САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
КАФЕДРА КОМПЬЮТЕРНОГО МОДЕЛИРОВАНИЯ И МНОГОПРОЦЕССОРНЫХ СИСТЕМ

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

Тема: «Оценка и визуализация оптического потока в задачах обработки изображений»

Направление: 02.04.02 – ФИиИТ

Магистерская программа: ВМ.5502.2014 – Вычислительные технологии

Выполнил студент гр. 633 Смирнов Константин Валерьевич

Научный руководитель,
к. т. н., доцент В. М. Гришкин

Санкт-Петербург – 2016

Содержание

Введение

В настоящее время широкий интерес вызывают задачи распознавания [1-9]. В данной работе рассматривается задача распознавания движения на изображении или на видео. Современные информационные технологии позволяют ускорить методы, которые применяются для решения данной задачи, например, за счет параллельной обработки различных частей видео или изображений из одной временной последовательности.

Целью данной работы является построение прототипа распределенной системы для распознавания движения на видео или последовательности изображений.

В качестве основы можно рассмотреть, семейство методов, основанных на представлении движения на изображении в форме поля скоростей или оптического потока [4-5, 10]. Поле скоростей (оптический поток) - одна из форм представления движения объектов на изображении, при котором каждой точке изображения ставится в соответствие вектор скорости, отвечающий соответствующей точке на объекте. При постановке задачи распознавания движения в форме поиска оптического потока существует несколько методов для ее решения.

1. Фазовая корреляция — основан на инверсия нормализованного перекрестного спектра.

2. Блочные методы — минимизация суммы квадратов или суммы модулей разностей.

3. Группа дифференциальных методов оценки оптического потока, основанных на частных производных:

3.1. Алгоритм Лукаса-Канаде — рассматриваются части изображения и аффинная модель движения.

3.2. Horn–Schunck — минимизация функционала, выражающего предположения постоянства цвета объектов на изображении и гладкости полу-

чаемого векторного поля.

3.3. Buxton–Buxton — основан на модели движения границ объектов в последовательности изображений.

4. Общие вариационные методы — модификации метода Horn-Schunck, использующие другие ограничения на данные и другие ограничения на гладкость.

5. Дискретные методы оптимизации — поисковое пространство квантуется, затем каждому пикселю изображения ставится в соответствие метка таким образом, чтобы расстояние между последовательными кадрами было минимальным. Оптимальное решение часто ищется с помощью алгоритмов нахождения минимального разреза и максимального потока в графе, линейного программирования или belief propagation.

В этой работе будут рассматриваться дифференциальные методы оценки поля скоростей, в его основе лежит вычисление частных производных по горизонтальному и вертикальному направлениям изображения. Как будет показано далее, одних только производных недостаточно чтобы определить смещения точек объектов на изображении. Поэтому существует множество модификаций названых алгоритмов, заточенных под определенные частные задачи.

Для построения прототипа распределенной системы для распознавания движения на видео или последовательности изображений в работе решаются следующие задачи:

1.Анализ существующих методов оценки поля скоростей, выбор основного алгоритма

2.Создание распределенной компьютерной реализации алгоритма вычисления поля скоростей на основе облачных технологий.

3.Анализ результатов.

Глава 1 Обзор существующих методов вычисления оптического потока

§1 Алгоритм Лукаса-Канаде

В данной главе рассматриваются несколько дифференциальных методов оценки поля скоростей. Начнем с рассмотрения самого простого из них алгоритма Лукаса-Канаде.

Будем считать, что рассматриваются черно-белые изображения, в этом случае цвет точки изображения определяется одним числом, которое будем далее называть яркостью.

В основе алгоритма Лукаса-Канаде лежит одно очень важное предположение: будем считать, что значения яркостей пикселей переходят из одного кадра в следующий без изменений. Таким образом, мы делаем допущение, что пиксели, относящиеся к одному и тому же объекту, могут сместиться в какую либо сторону, но их значение останется неизменным. Конечно, это предположение слабо связано с реальностью, так как от кадра к кадру могут меняться глобальные условия освещения и освещенность самого движущегося объекта. Это допущение в дальнейшем приводит к некоторым проблемам, но, не смотря на это, оно достаточно хорошо работает на практике.

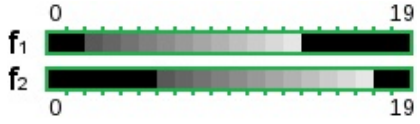

Рис. 1: Пример линейных изображений.

На математическом языке это допущение можно записать так: $I(x, y, t) = I(x + u_x, y + u_y, t + 1)$. Где I — это функция яркости пикселей от положения

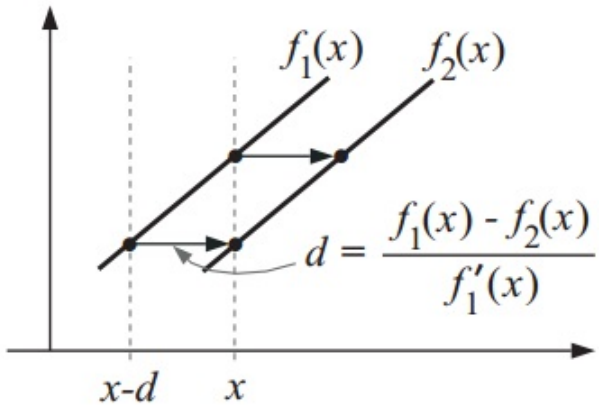

Рис. 2: Линейные изображения как функции.

на кадре и времени. Другими словами x и y — это вещественные координаты точки в плоскости кадра, u_x и u_y — это смещение, а t — это номер кадра в последовательности. Условимся, что между двумя соседними кадрами проходит единичный отрезок времени.

Для начала рассмотрим одномерный случай. Представим себе два одномерных кадра 1 пиксель в высоту и 20 пикселей в ширину. На втором кадре изображение немного смещено вправо. Именно это смещение мы и хотим найти. Для этого представим эти же кадры в виде функций. На входе позиция пикселя, на выходе — его интенсивность. В таком представление искомое смещение d видно еще более наглядно. В соответствии с нашим предположением, f_2 это просто смещенная f_1 , то есть можем сказать, что $f_2 = I(x - d)$.

Обратим внимание, что f_1 и f_2 при желании можно записать и в общем виде: $f_1(x) = I(x, y, t)$, $f_2(x) = I(x, y, t)$ где y и t зафиксированы и равны нулю.

Для каждой координаты нам известны значения f_1 и f_2 в этой точке, кроме того мы можем вычислить их производные. Свяжем известные значения со смещением d. Для этого запишем разложение в ряд Тейлора для $f_1(x - d)$:

$$f_1(x-d) = f_1(x) - df_1'(x) + O(d^2 f_1'')$$

Сделаем второе важное предположение: Предположим, что $f_1(x - d)$ достаточно хорошо аппроксимируется первой производной. Сделав это предположение, отбросим всё что после первой производной:

$$f_1(x-d) = f_1(x) - df_1'(x)$$

Насколько это корректно? В общем-то, не очень, тут мы теряем в точности, если только наша функция/изображение не строго линейна, как в нашем искусственном примере. Зато это существенно упрощает метод, а для достижения требуемой точности можно сделать последовательное приближение, которое мы рассмотрим позже.

Мы почти у цели. Смещение d — это наша искомая величина, поэтому надо что-то сделать с $f_1(x-d)$. Как мы условились ранее, $f_2(x) = f_1(x-d)$, поэтому просто перепишем:

$$f_2(x) = f_1(x) - df_1'(x)$$

То есть:

$$d = \frac{f_1(x) - f_2(x)}{f_1'(x)}$$

Теперь перейдем от одномерного случая к двумерному. Запишем разложение в ряд Тейлора для $I(x+u_x, y+u_y, t)$ и сразу отбросим все старшие производные. Вместо первой производной появляется градиент:

$$I(x+u_x, y+u_y, t) = I(x,y,t) + \mathbf{u}\nabla I(x,y,t)$$

Где $\mathbf{u} = (u_x, u_y)^T$ — вектор смещения. В соответствии со сделанным допущением $I(x,y,t) = I(x+u_x, y+u_y, t+1)$. Обратите внимание, что это

выражение эквивалентно $I(x+u_x, y+u_y, t) = I(x, y, t+1)$. Это то, что нам нужно. Перепишем:

$$I(x, y, t+1) = I(x, y, t) + \mathbf{u}\nabla I(x, y, t)$$

$$I(x, y, t) - I(x, y, t+1) + \mathbf{u}\nabla I(x, y, t) = 0$$

Поскольку между двумя кадрами проходит единичный интервал времени, то можно сказать, что $I(x, y, t) - I(x, y, t+1)$ есть не что иное, как производная по времени. Перепишем:

$$\frac{\partial I(x, y, t)}{\partial t} + \mathbf{u}\nabla I(x, y, t) = 0$$

Перепишем ещё раз, раскрыв градиент:

$$\frac{\partial I(x, y, t)}{\partial t} + u_x \frac{\partial I(x, y, t)}{\partial x} + u_y \frac{\partial I(x, y, t)}{\partial y} = 0$$

Мы получили уравнение, которое говорит нам о том, что сумма частных производных должны быть равна нулю. Проблема только в том, что уравнение у нас одно, а неизвестных в нем два: u_x и u_y. На этом моменте начинается полет фантазии и разнообразие подходов.

Сделаем третье предположение: Предположим, что соседние пиксели смещаются на одинаковое расстояние. Возьмем фрагмент изображения, скажем 5 на 5 пикселей, и условимся, что для каждого из 25 пикселей u_x и u_y равны. Тогда вместо одного уравнения мы получим сразу 25 уравнений! Очевидно, что в общем случае система не имеет решения, поэтому будем искать такие u_x и u_y, которые минимизируют ошибку:

$$E(u_x, u_y) = \sum_{i,j} g(x_i, y_j) \left[\frac{\partial I(x_i, y_j, t)}{\partial t} + u_x \frac{\partial I(x_i, y_j, t)}{\partial x} + u_y \frac{\partial I(x_i, y_j, t)}{\partial y} \right]^2$$

Здесь g — это функция, определяющая весовые коэффициенты для пикселей. Самые распространенный вариант — двухмерная гауссиана, которая дает наибольший вес центральному пикселю и все меньший по мере удаления от центра.

Чтобы найти минимум $E(u_x, u_y)$ воспользуемся методом наименьших квадратов, найдем её частные производные по u_x и u_y:

$$\frac{\partial E(u_x, u_y)}{\partial u_x} =$$

$$= 2\sum_{i,j} g(x_i, y_j) \left[\frac{\partial I(x_i, y_j, t)}{\partial t} + u_x \frac{\partial I(x_i, y_j, t)}{\partial x} + u_y \frac{\partial I(x_i, y_j, t)}{\partial y}\right] \frac{\partial I(x_i, y_j, t)}{\partial x} =$$

$$= 2\sum_{i,j} g(x_i, y_j)[u_x \left(\frac{\partial I(x_i, y_j, t)}{\partial x}\right)^2 + u_y \frac{\partial I(x_i, y_j, t)}{\partial y} \frac{\partial I(x_i, y_j, t)}{\partial x} +$$

$$+ \frac{\partial I(x_i, y_j, t)}{\partial t} \frac{\partial I(x_i, y_j, t)}{\partial x}]$$

$$\frac{\partial E(u_x, u_y)}{\partial u_y} =$$

$$= 2\sum_{i,j} g(x_i, y_j) \left[\frac{\partial I(x_i, y_j, t)}{\partial t} + u_x \frac{\partial I(x_i, y_j, t)}{\partial x} + u_y \frac{\partial I(x_i, y_j, t)}{\partial y}\right] \frac{\partial I(x_i, y_j, t)}{\partial y} =$$

$$= 2\sum_{i,j} g(x_i, y_j)[u_x \frac{\partial I(x_i, y_j, t)}{\partial x} \frac{\partial I(x_i, y_j, t)}{\partial y} + u_y \left(\frac{\partial I(x_i, y_j, t)}{\partial y}\right)^2 +$$

$$+ \frac{\partial I(x_i, y_j, t)}{\partial t} \frac{\partial I(x_i, y_j, t)}{\partial y}]$$

Перепишем в более компактной форме и приравняем к нулю:

$$\frac{\partial E(u_x, u_y)}{\partial u_x} = \sum_{i,j} g(x_i, y_j) \left[u_x \left(\frac{\partial I}{\partial x}\right)^2 + u_y \frac{\partial I}{\partial y} \frac{\partial I}{\partial x} + \frac{\partial I}{\partial t} \frac{\partial I}{\partial x}\right]$$

$$\frac{\partial E(u_x, u_y)}{\partial u_y} = \sum_{i,j} g(x_i, y_j) \left[u_x \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} + u_y \left(\frac{\partial I}{\partial y}\right)^2 + \frac{\partial I}{\partial t} \frac{\partial I}{\partial y}\right]$$

Перепишем эти два уравнения в матричной форме:

$$M\mathbf{u} = \mathbf{b}$$

Где

$$M = \begin{bmatrix} \sum_{i,j} g(x_i, y_j) \left(\frac{\partial I}{\partial x}\right)^2 & \sum_{i,j} g(x_i, y_j) \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \\ \sum_{i,j} g(x_i, y_j) \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} & \sum_{i,j} g(x_i, y_j) \left(\frac{\partial I}{\partial y}\right)^2 \end{bmatrix}$$

$$\mathbf{b} = - \begin{bmatrix} \sum_{i,j} g(x_i, y_j) \frac{\partial I}{\partial t} \frac{\partial I}{\partial x} \\ \sum_{i,j} g(x_i, y_j) \frac{\partial I}{\partial t} \frac{\partial I}{\partial y} \end{bmatrix}$$

$$\mathbf{u} = - \begin{bmatrix} u_x \\ u_y \end{bmatrix}$$

Если матрица обратима (имеет ранг 2), можем вычислить u_x и u_y, которые минимизируют ошибку E:

$$\widetilde{u} = M^{-1}\mathbf{b}$$

Мы знаем приблизительное смещение пикселей между двумя соседними кадрами.

Поскольку в нахождении смещения каждого пикселя участвуют также соседние с ним пиксели, при реализации данного метода целесообразно предварительно посчитать производные кадра по горизонтали и вертикали.

Описанный выше метод основан на трех значительных допущениях, которые с одной стороны дают нам принципиальную возможность определить оптический поток, но с другой стороны вносят погрешность. Хорошая новость для перфекционистов состоит в том, что одно допущение нужно нам только для упрощения метода, и с его последствиями мы можем бороться. Мы предполагали, что для аппроксимации смещения нам будет достаточно первой производной. В общем случае это конечно же не так (рисунок слева). Для достижение требуемой точности смещение для каждой пары кадров (назовём их F_i и F_{i+1}) можно вычислять итеративно. В

литературе это называется искажением (warping). На практике это означает, что, вычислив смещения на первой итерации, мы перемещаем каждый пиксель кадра F_{i+1} в противоположную сторону так, чтобы это смещение компенсировать. На следующей итерации вместо исходного кадра F_{i+1} мы будем использовать его искаженный вариант F'_{i+1}. И так далее, пока на очередной итерации все полученные смещения не окажутся меньше заданного порогового значения. Итоговое смещение для каждого конкретного пикселя мы получаем как сумму его смещений на всех итерациях.

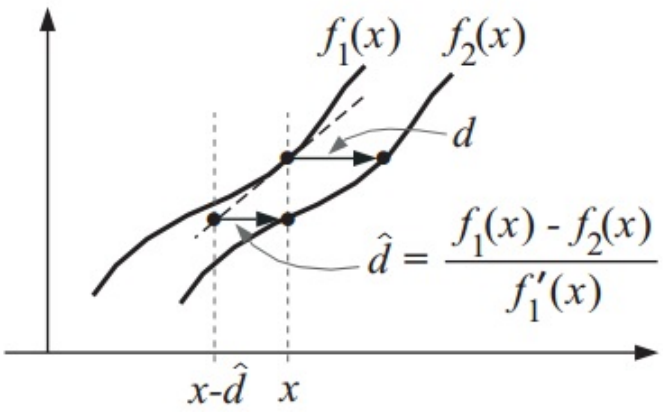

Рис. 3: Пример линейных изображений.

По своей природе данный метод является локальным, то есть при определении смещения конкретного пикселя принимается во внимание только область вокруг этого пикселя — локальная окрестность. Как следствие, невозможно определить смещения внутри достаточно больших (больше размера локальной окрестности) равномерно окрашенных участков кадра. К счастью на реальных кадрах такие участки встречаются не часто, но эта особенность все же вносит дополнительное отклонение от истинного смещения.

Ещё одна проблема связана с тем, что некоторые текстуры в изображении дают вырожденную матрицу , для которой не может быть найдена обратная матрица. Соответственно, для таких текстур мы не сможем

определить смещение. То есть движение вроде есть, но непонятно в какую сторону. В общем-то, от этой проблемы страдает не только рассмотренный метод. Даже глаз человека воспринимает такое движение не однозначно.

§2 Алгоритм Horn-Schunck

Следующий дифференциальный метод оценки поля скоростей, который будет нами рассмотрен — алгоритм Horn-Schunck

Будем использовать введенные ранее обозначения области в пространстве плоскости изображений и времени $\Omega \subset R^3$ и подобласти, отвечающей моменту времени t_0 Ω_{t_0}. Плоскость изображений - модель экрана, на который проецируется движение. Экран по форме с течением времени не меняется, следовательно $\Omega_{t_1} = \Omega_{t_2}, \forall t_1, t_2 \in R$. Проецирование будем полагать ортогональным. Если наблюдатель перемещается в пространстве, то эта плоскость будет перемещаться вместе с ним, следовательно, при отсутствии движения объектов в пространстве на плоскости изображений будет просматриваться движение объектов, обратное движению наблюдателя.

Внешние условия, отвечающие окружающей среде, безусловно, будут влиять на изображение движения. Мы будем определять проекцию движения, на плоскость изображений, так как у нас нет данных о движении, ортогональном этой плоскости.

Основной характеристикой, которой мы будем пользоваться, будет являться яркость точек изображений. На нее кроме текстуры передвигающихся объектов будет влиять освещенность.

В самом простом случае, можно предположить постоянство освещения, чему будет отвечать постоянство яркости, это можно выразить следующим образом:

$$I(\mathbf{x} + \mathbf{u}) - I(\mathbf{x}) \xrightarrow{\mathbf{u}} 0, \mathbf{x}, \mathbf{u} \in \Omega_{t_0}.$$

Если освещение изменяется равномерно по всему пространству, т.е. по всей плоскости изображений, то мы можем перейти к градиенту яркости чтобы рассмотреть рельефность изображения и записать это в виде

$$\bigtriangledown I(\mathbf{x}+\mathbf{u}) - \bigtriangledown I(\mathbf{x}) \overset{\mathbf{u}}{\to} 0, \mathbf{x}, \mathbf{u} \in \Omega_{t_0}.$$

Если учитывать возможность локального изменения освещенности, то изображение рельефности изображения может искажаться, при точечной модели освещения и условии равномерного распространения цвета, это искажение будет только в форме круга, чем дальше от центра которого тем будет меньше искажение. Переход к матрицам Гессе для яркости позволяет учитывать такую модель, это можно записать в виде:

$$I_{x_i x_j}(\mathbf{x}+\mathbf{u}) - I_{x_i x_j}(\mathbf{x}) \overset{\mathbf{u}}{\to} 0, i, j = 1, 2, \mathbf{x}, \mathbf{u} \in \Omega_{t_0}.$$

Для большей эффективности, эти условия можно комбинировать вместе

Для ввода зависимости между соседними точками плоскости изображений введем в качестве ограничения гладкость потока по переменным x_1, x_2, для этого будем минимизировать разницу скоростей соседних точек на плоскости изображений. С помощью градиента потока это можно выразить в следующем виде:

$$\| \bigtriangledown u_1\|^2 + \| \bigtriangledown u_2\|^2 \overset{\mathbf{u}}{\to} 0, \mathbf{u} \in \Omega_{t_0}.$$

Здесь и далее в этой главе будет рассматриваться Евклидова норма матрицы

$$\|A\| = \sqrt{\sum_{i,j=1,1}^{(n,m)} a_{i,j}}, \text{где матрица} A = \{a_{i,j}\}_{i=1\ldots n, j=1\ldots m}.$$

Далее рассмотрим формализацию для пар условий постоянства яркости и гладкости потока, условий постоянства Гессиана яркости и гладкости потока.

Запишем условия и в интегральной форме для всех точек плоскости изображений в момент времени t_0:

$$\int_{\Omega_{t_0}} ([I(\mathbf{x}+\mathbf{u}) - I(\mathbf{x})]^2) dx_1 dx_2 + \alpha \int_{\Omega_{t_0}} (\| \bigtriangledown u_1\|^2 + \| \bigtriangledown u_2\|^2) dx_1 dx_2 \xrightarrow{\mathbf{u}} 0.$$

Здесь $\alpha > 0$ - параметр регуляризации. В общем случае для произвольной последовательности изображений нам ничего о них неизвестно, поэтому будем полагать граничные условия нулевыми.

Переходя к уравнениям Эйлера-Лагранжа, мы получим

$$\begin{cases} I_{x_1}(\mathbf{x}+\mathbf{u})[I(\mathbf{x}+\mathbf{u}) - I(\mathbf{x})] + \alpha \bigtriangledown^2 u_1 = 0 \\ I_{x_2}(\mathbf{x}+\mathbf{u})[I(\mathbf{x}+\mathbf{u}) - I(\mathbf{x})] + \alpha \bigtriangledown^2 u_2 = 0 \end{cases}.$$

Далее, производя аппроксимацию по формулам Тейлора, придем к следующей системе

$$\begin{cases} I_{x_1}(\mathbf{x})[I_t(\mathbf{x}) + I_{x_1}(\mathbf{x})u_1 + I_{x_2}(\mathbf{x})u_2] - \alpha \bigtriangledown^2 u_1 = 0 \\ I_{x_2}(\mathbf{x})[I_t(\mathbf{x}) + I_{x_1}(\mathbf{x})u_1 + I_{x_2}(\mathbf{x})u_2] - \alpha \bigtriangledown^2 u_2 = 0 \end{cases}.$$

К дискретной форме перейдем следующим образом. Положим u_{1ij} и u_{2ij} как проекции скоростей по осям в точке с координатами i, j, а x_{1ij} и x_{2ij} как координаты соответствующей точки $\mathbf{x}_{ij}$. $i \in 1..N-1$, $j \in 1..M-1$. Для границ $i = 0, i = N, j = 0, j = M$ положим u_{1ij}и u_{2ij} равным граничным условиям, задаваемым изначально, обычно они берутся равными нулю. Раскроем лапласиан в точке с координатами i, j по следующей формуле:

$$\bigtriangledown^2 u_{1ij} = u_{1i+1j} + u_{1i-1j} + u_{1ij-1} + u_{1ij-1} - 4u_{1ij}$$

$$\bigtriangledown^2 u_{2ij} = u_{2i+1j} + u_{2i-1j} + u_{2ij-1} + u_{2ij-1} - 4u_{2ij}$$

Тогда система уравнений принимает вид линейной системы уравне-

ний:

$$\begin{cases} I_{x_1}(\mathbf{x}_{ij})[I_t(\mathbf{x}_{ij}) + I_{x_1}(\mathbf{x}_{ij})u_{1ij} + I_{x_2}(\mathbf{x}_{ij})u_{2ij}] - \\ -\frac{1}{4}\alpha(u_{1i+1j} + u_{1i-1j} + u_{1ij-1} + u_{1ij-1} - 4u_{1ij}) = 0 \\ I_{x_2}(\mathbf{x}_{ij})[I_t(\mathbf{x}_{ij}) + I_{x_1}(\mathbf{x}_{ij})u_{1ij} + I_{x_2}(\mathbf{x}_{ij})u_{2ij}] - \\ -\frac{1}{4}\alpha(u_{2i+1j} + u_{2i-1j} + u_{2ij-1} + u_{2ij-1} - 4u_{2ij}) = 0 \end{cases}.$$

Введем обозначения:

$$A_{11ij} = I_{x_1}(\mathbf{x}_{ij})I_{x_1}(\mathbf{x}_{ij}), A_{12ij} = I_{x_1}(\mathbf{x}_{ij})I_{x_2}(\mathbf{x}_{ij}), A_{22ij} = I_{x_2}(\mathbf{x}_{ij})I_{x_2}(\mathbf{x}_{ij}),$$
$$B_{1ij} = I_{x_1}(\mathbf{x}_{ij})I_t(\mathbf{x}_{ij}), B_{2ij} = I_{x_1}(\mathbf{x}_{ij})I_t(\mathbf{x}_{ij}).$$

Сгруппируем u_{1ij} и u_{2ij}:

$$\begin{cases} (A_{11ij} + \alpha)u_{1ij} + A_{12ij}u_{1ij} = \alpha(u_{1i+1j} + u_{1i-1j} + u_{1ij-1} + u_{1ij-1}) - B_{1ij} \\ A_{12ij}u_{2ij} + (A_{22ij} + \alpha)u_{2ij} = \alpha(u_{2i+1j} + u_{2i-1j} + u_{2ij-1} + u_{2ij-1}) - B_{2ij} \end{cases}.$$

Слева получилась невырожденная матрица, так как определитель:

$$(A_{11ij} + \alpha)(A_{22ij} + \alpha) - A_{12ij}A_{12ij} =$$

$$= I_{x_1}(\mathbf{x}_{ij})I_{x_1}(\mathbf{x}_{ij})I_{x_2}(\mathbf{x}_{ij})I_{x_2}(\mathbf{x}_{ij}) + (I_{x_1}(\mathbf{x}_{ij})I_{x_1}(\mathbf{x}_{ij}) + I_{x_2}(\mathbf{x}_{ij})I_{x_2}(\mathbf{x}_{ij})) +$$
$$+\alpha^2 - I_{x_1}(\mathbf{x}_{ij})I_{x_1}(\mathbf{x}_{ij})I_{x_2}(\mathbf{x}_{ij})I_{x_2}(\mathbf{x}_{ij}) =$$
$$= \left(I_{x_1}(\mathbf{x}_{ij})^2 + I_{x_2}(\mathbf{x}_{ij})^2\right) + \alpha^2 > 0$$

Будем решать систему итерационным методом:

$$\begin{cases} (A_{11ij} + \alpha)u^k_{1ij} + A_{12ij}u^k_{1ij} = \alpha(u^{k-1}_{1i+1j} + u^{k-1}_{1i-1j} + u^{k-1}_{1ij-1} + u^{k-1}_{1ij-1}) - B_{1ij} \\ A_{12ij}u^k_{2ij} + (A_{22ij} + \alpha)u^k_{2ij} = \alpha(u^{k-1}_{2i+1j} + u^{k-1}_{2i-1j} + u^{k-1}_{2ij-1} + u^{k-1}_{2ij-1}) - B_{2ij} \end{cases}.$$

Пусть:

$$\widetilde{u}^{k-1}_1 = u^{k-1}_{1i+1j} + u^{k-1}_{1i-1j} + u^{k-1}_{1ij-1} + u^{k-1}_{1ij-1}$$
$$\widetilde{u}^{k-1}_2 = u^{k-1}_{2i+1j} + u^{k-1}_{2i-1j} + u^{k-1}_{2ij-1} + u^{k-1}_{2ij-1}$$

Тогда окончательное решение системы линейных уравнений запишется в виде:

$$\begin{cases} u^k_{1ij} = \widetilde{u}^{k-1}_1 - (A_{11ij} + A_{22ij} + \alpha^2)^{-1}(A_{11ij}\widetilde{u}^{k-1}_1 + A_{12ij}\widetilde{u}^{k-1}_2 + B_{1ij}) \\ u^k_{2ij} = \widetilde{u}^{k-1}_2 - (A_{11ij} + A_{22ij} + \alpha^2)^{-1}(A_{12ij}\widetilde{u}^{k-1}_1 + A_{22ij}\widetilde{u}^{k-1}_2 + B_{2ij}) \end{cases}.$$

При $k = 0$ u_{1ij} и u_{2ij} полагаются равным начальному приближению, рационально брать это приближение равным нулю или равным результату, полученным для предыдущего момента t.

§3 Обобщения для предположений постоянства производных

Приведенный в предыдущем параграфе алгоритм можно обобщить на случаи рассмотрения производных от функции $I(x, y, t)$. На практике это имеет смысл, если цвет объекта меняется с течением времени и предположение постоянства яркости теряет смысл. Например, это возможно при изменении освещенности сцены с течением времени. При одинаковом изменении освещенности с течением времени имеет смысл рассматривать вместо постоянства яркости $I(x, y, t) = I(x + u_x, y + u_y, t + 1)$ постоянство ее производных:

$$\frac{\partial I(x,y,t)}{\partial x} = \frac{\partial I(x+u_x, y+u_y, t+1)}{\partial x}$$

$$\frac{\partial I(x,y,t)}{\partial y} = \frac{\partial I(x+u_x, y+u_y, t+1)}{\partial y}$$

Если освещенность изменяется в плоскости изображения с течением времени не равномерно, т.е. при наличии динамических источников света, то имеет смысл рассматривать предположения постоянства производных более высокого порядка: постоянство гессиана, лапласиана, определителя матрицы Гессе:

$$H(I(x,y,t)) = H(I(x+u_x, y+u_y, t+1))$$

$$\frac{\partial^2 I(x,y,t)}{\partial x^2} + \frac{\partial^2 I(x,y,t)}{\partial y^2} =$$

$$= \frac{\partial^2 I(x+u_x, y+u_y, t+1)}{\partial x^2} + \frac{\partial^2 I(x+u_x, y+u_y, t+1)}{\partial y^2}$$

$$det(H(I(x,y,t)) = det(H(I(x+u_x, y+u_y, t+1))$$

где

$$H(I(x,y,t) = \begin{bmatrix} \frac{\partial^2 I(x+u_x, y+u_y, t+1)}{\partial x^2} & \frac{\partial^2 I(x+u_x, y+u_y, t+1)}{\partial x \partial y} \\ \frac{\partial^2 I(x+u_x, y+u_y, t+1)}{\partial x \partial y} & \frac{\partial^2 I(x+u_x, y+u_y, t+1)}{\partial y^2} \end{bmatrix}$$

В случае постоянства гессиана обобщение можно сделать следующим образом:

$$\int_{\Omega_{t_0}} \left(\sum_{n,m=1,1}^{(2,2)} \left(I_{x_n x_m}(\mathbf{x}+\mathbf{u}) - I_{x_n x_m}(\mathbf{x})\right)^2 \right) dx_1 dx_2 +$$

$$+\alpha \int_{\Omega_{t_0}} (\| \bigtriangledown u_1\|^2 + \| \bigtriangledown u_2\|^2) dx_1 dx_2 \xrightarrow{\mathbf{u}} 0.$$

Здесь $\alpha > 0$ - параметр регуляризации. В общем случае для произвольной последовательности изображений нам ничего о них неизвестно, поэтому будем полагать граничные условия нулевыми.

Переходя к уравнениям Эйлера-Лагранжа, мы получим

$$\begin{cases} \sum_{n,m=1,1}^{(2,2)} \left(I_{x_n x_m x_1}(\mathbf{x}+\mathbf{u})[(I_{x_n x_m}(\mathbf{x}+\mathbf{u}) - I_{x_n x_m}(\mathbf{x})]\right) + \\ \qquad +\alpha \bigtriangledown^2 u_1 = 0 \\ \sum_{n,m=1,1}^{(2,2)} \left(I_{x_n x_m x_2}(\mathbf{x}+\mathbf{u})[(I_{x_n x_m}(\mathbf{x}+\mathbf{u}) - I_{x_n x_m}(\mathbf{x})]\right) + \\ \qquad +\alpha \bigtriangledown^2 u_2 = 0 \end{cases}.$$

Далее, производя аппроксимацию по формулам Тейлора, придем к следующей системе

$$\begin{cases} \sum_{n,m=1,1}^{(2,2)} \left(I_{x_n x_m x_1}(\mathbf{x})[I_{x_n x_m t}(\mathbf{x}) + I_{x_n x_m x_1}(\mathbf{x})u_1 + I_{x_n x_m x_2}(\mathbf{x})u_2]\right) + \\ \qquad +\alpha \bigtriangledown^2 u_1 = 0 \\ \sum_{n,m=1,1}^{(2,2)} \left(I_{x_n x_m x_2}(\mathbf{x})[I_{x_n x_m t}(\mathbf{x}) + I_{x_n x_m x_1}(\mathbf{x})u_1 + I_{x_n x_m x_2}(\mathbf{x})u_2]\right) + \\ \qquad +\alpha \bigtriangledown^2 u_2 = 0 \end{cases}.$$

Эту систему можно решать тем же образом, который рассматривался для решения системы в предыдущем параграфе, только матрицы M теперь

будет записываться следующим образом:

$$
\begin{aligned}
A_{11ij} &= \sum_{n,m=1,1}^{(2,2)} I_{x_n x_m x_1}(\mathbf{x}_{ij}) I_{x_n x_m x_1}(\mathbf{x}_{ij}),\\
A_{11ij} &= \sum_{n,m=1,1}^{(2,2)} I_{x_n x_m x_1}(\mathbf{x}_{ij}) I_{x_n x_m x_2}(\mathbf{x}_{ij}),\\
A_{11ij} &= \sum_{n,m=1,1}^{(2,2)} I_{x_n x_m x_2}(\mathbf{x}_{ij}) I_{x_n x_m x_2}(\mathbf{x}_{ij}),\\
B_{1ij} &= \sum_{n,m=1,1}^{(2,2)} I_{x_n x_m x_1}(\mathbf{x}_{ij}) I_{x_n x_m t}(\mathbf{x}_{ij}),\\
B_{2ij} &= \sum_{n,m=1,1}^{(2,2)} I_{x_n x_m x_2}(\mathbf{x}_{ij}) I_{x_n x_m t}(\mathbf{x}_{ij}).
\end{aligned}
$$

§4 Детали дискретизации

Рассмотрим случай предположения постоянства яркости точки и более подробно опишем, как можно осуществлять переход от системы дифференциальных уравнений к системе алгебраических уравнений.

Есть система дифференциальных уравнений:

$$\begin{cases} I_{x_1}(\mathbf{x})[I_t(\mathbf{x}) + I_{x_1}(\mathbf{x})u_1 + I_{x_2}(\mathbf{x})u_2] - \alpha \bigtriangledown^2 u_1 = 0 \\ I_{x_2}(\mathbf{x})[I_t(\mathbf{x}) + I_{x_1}(\mathbf{x})u_1 + I_{x_2}(\mathbf{x})u_2] - \alpha \bigtriangledown^2 u_2 = 0 \end{cases}.$$

Как и раньше, положим u_{1ij} и u_{2ij} как проекции скоростей по осям в точке с координатами i, j, а x_{1ij} и x_{2ij} как координаты соответствующей точки $\mathbf{x}_{ij}$. $i \in 1..N-1$, $j \in 1..M-1$. Для границ $i = 0, i = N, j = 0, j = M$ положим u_{1ij}и u_{2ij} равным граничным условиям, задаваемым изначально, обычно они берутся равными нулю.

Возможно несколько способов раскрытия лапласиана. Рассматриваемый ранее:

$$\bigtriangledown^2 u_{1ij} \approx u_{1i+1j} + u_{1i-1j} + u_{1ij-1} + u_{1ij-1} - 4u_{1ij}$$

$$\bigtriangledown^2 u_{2ij} \approx u_{2i+1j} + u_{2i-1j} + u_{2ij-1} + u_{2ij-1} - 4u_{2ij}$$

Оценка производных функции $I(x, y, t)$ тоже может быть осуществлена несколькими способами:

$$\frac{\partial I_(x_{ij})}{\partial x} \approx \frac{1}{2}(I(x_{i+1j}) - I(x_{i-1j}))$$

$$\frac{\partial I(x_{ij})}{\partial x} \approx I(x_{ij}) - I(x_{i-1j})$$

$$\frac{\partial I(x_{ij})}{\partial x} \approx I(x_{i+1j}) - I(x_{ij})$$

Анализ точности этих формул можно найти в [14]. Возможно использование более сложных формул, которые требуют большего количества точек, но при оценке производных по времени это тесно связано с количеством доступных изображений.

Использование более сложных конструкций приводит к размытию оригинальной производной.

Например, рассмотрим движение черного прямоугольника на белом фоне вдоль оси y. Пусть момент времени $t+1$ отличается от момента времени t сдвигом прямоугольника на один пиксель изображения. Таким образом, ожидается наличие производной по оси y не равной нулю только для двух фиксированных значений x, соответствующих верхней и нижней грани прямоугольника. При использовании первой из трех указанных разностных схем происходит размытие производной вдоль оси y, движению верхней грани будет соответствовать не одна ненулевая производная, а две в два раза меньшие оригинальной.

В оригинальной статье для метода Horn-Schunk используется оценка производных с усреднением:

$$\frac{\partial I(x_{ijk})}{\partial x} \approx \frac{1}{4}(I(x_{i+1jk}) - I(x_{ijk}) + I(x_{i+1j+1k}) - I(x_{ij+1k}) +$$
$$+ I(x_{i+1jk+1}) - I(x_{ijk+1}) + I(x_{i+1j+1k+1}) - I(x_{ij+1k+1}))$$
$$\frac{\partial I(x_{ijk})}{\partial y} \approx \frac{1}{4}(I(x_{ij+1k}) - I(x_{ijk}) + I(x_{i+1j+1k}) - I(x_{i+1jk}) +$$
$$+ I(x_{ij+1k+1}) - I(x_{ijk+1}) + I(x_{i+1j+1k+1}) - I(x_{i+1jk+1}))$$
$$\frac{\partial I(x_{ijk})}{\partial t} \approx \frac{1}{4}(I(x_{ijk+1}) - I(x_{ijk}) + I(x_{ij+1k+1}) - I(x_{ij+1k}) +$$
$$+ I(x_{i+1jk+1}) - I(x_{i+1jk}) + I(x_{i+1j+1k+1}) - I(x_{i+1j+1k}))$$

где k — дискретный момент времени, при разбиении для имеющихся изображений. Такой подход дополнительно размывает изображение.

Глава 2 Алгоритм построения поля скоростей

§1 Смешаный алгоритм Лукаса-Канаде и Horn Schunck

В рамках данной работы предлагается модификация алгоритма Horn-Schunck, которая заключается в объединении оригинального алгоритма Horn-Schunck и алгоритма Лукаса-Канаде, и модификация выделения границ. Далее в этом параграфе приводится математическая формализация этого смешанного алгоритма.

Будем использовать введенные ранее в описании алгоритма Horn Schuncd обозначения.

Ранее использовалось условие постоянства яркости в точке:

$$I(\mathbf{x}+\mathbf{u}) - I(\mathbf{x}) \stackrel{\mathbf{u}}{\to} 0, \mathbf{x}, \mathbf{u} \in \Omega_{t_0}.$$

Теперь заменим его на следующие r условий:

$$I(\mathbf{x}^{[d]}+\mathbf{u}) - I(\mathbf{x}^{[d]}) \stackrel{\mathbf{u}}{\to} 0, \mathbf{x}^{[d]}, \mathbf{u} \in \Omega_{t_0}.$$

Здесь множество точек $\{\mathbf{x}^{[d]}, d = 1...r\}$ - точки из окрестности точки $\mathbf{x}$. Запишем это условие вместе с условием гладкости потока для всех точек плоскости изображений в момент времени t_0:

$$\int_{\Omega_{t_0}} \sum_{d=1}^{r} (\left[I(\mathbf{x}^{[d]}+\mathbf{u}) - I(\mathbf{x}^{[d]})\right]^2) dx_1 dx_2 + \alpha \int_{\Omega_{t_0}} (\|\bigtriangledown u_1\|^2 + \|\bigtriangledown u_2\|^2) dx_1 dx_2 \stackrel{\mathbf{u}}{\to} 0.$$

Здесь $\alpha > 0$ - параметр регуляризации. В общем случае для произвольной последовательности изображений нам ничего о них неизвестно, поэтому будем полагать граничные условия нулевыми.

Переходя к уравнениям Эйлера-Лагранжа, мы получим

$$\begin{cases} \sum_{d=1}^{r} I_{x_1}(\mathbf{x}^{[d]} + \mathbf{u})[I(\mathbf{x}^{[d]} + \mathbf{u}) - I(\mathbf{x}^{[d]})] + \alpha \bigtriangledown^2 u_1 = 0 \\ \sum_{d=1}^{r} I_{x_2}(\mathbf{x}^{[d]} + \mathbf{u})[I(\mathbf{x}^{[d]} + \mathbf{u}) - I(\mathbf{x}^{[d]})] + \alpha \bigtriangledown^2 u_2 = 0 \end{cases}.$$

Далее, производя аппроксимацию по формулам Тейлора, придем к следующей системе

$$\begin{cases} \sum_{d=1}^{r} I_{x_1}(\mathbf{x}^{[d]})[I_t(\mathbf{x}^{[d]}) + I_{x_1}(\mathbf{x}^{[d]})u_1 + I_{x_2}(\mathbf{x}^{[d]})u_2] - \alpha \bigtriangledown^2 u_1 = 0 \\ \sum_{d=1}^{r} I_{x_2}(\mathbf{x}^{[d]})[I_t(\mathbf{x}^{[d]}) + I_{x_1}(\mathbf{x}^{[d]})u_1 + I_{x_2}(\mathbf{x}^{[d]})u_2] - \alpha \bigtriangledown^2 u_2 = 0 \end{cases}.$$

К дискретной форме перейдем следующим образом. Положим u_{1ij} и u_{2ij} как проекции скоростей по осям в точке с координатами i, j, а $x_{1ij}^{[d]}$ и $x_{2ij}^{[d]}$ как координаты соответствующей точки $\mathbf{x}_{ij}^{[d]}$. $i \in 1..N-1$, $j \in 1..M-1$. Для границ $i = 0, i = N, j = 0, j = M$ положим u_{1ij}и u_{2ij} равным граничным условиям, задаваемым изначально, обычно они берутся равными нулю. Раскроем лапласиан в точке с координатами i, j по следующей формуле:

$$\bigtriangledown^2 u_{1ij} = u_{1i+1j} + u_{1i-1j} + u_{1ij-1} + u_{1ij-1} - 4u_{1ij}$$

$$\bigtriangledown^2 u_{2ij} = u_{2i+1j} + u_{2i-1j} + u_{2ij-1} + u_{2ij-1} - 4u_{2ij}$$

Тогда система уравнений принимает вид линейной системы уравнений:

$$\begin{cases} \sum_{d=1}^{r} I_{x_1}(\mathbf{x}_{ij}^{[d]})[I_t(\mathbf{x}_{ij}^{[d]}) + I_{x_1}(\mathbf{x}_{ij}^{[d]})u_{1ij} + I_{x_2}(\mathbf{x}_{ij}^{[d]})u_{2ij}] - \\ \quad -\frac{1}{4}\alpha(u_{1i+1j} + u_{1i-1j} + u_{1ij-1} + u_{1ij-1} - 4u_{1ij}) = 0 \\ \sum_{d=1}^{r} I_{x_2}(\mathbf{x}_{ij}^{[d]})[I_t(\mathbf{x}_{ij}^{[d]}) + I_{x_1}(\mathbf{x}_{ij}^{[d]})u_{1ij} + I_{x_2}(\mathbf{x}_{ij}^{[d]})u_{2ij}] - \\ \quad -\frac{1}{4}\alpha(u_{2i+1j} + u_{2i-1j} + u_{2ij-1} + u_{2ij-1} - 4u_{2ij}) = 0 \end{cases}.$$

Введем обозначения:

$$A_{11ij} = \sum_{d=1}^{r} I_{x_1}(\mathbf{x}_{ij}^{[d]}) I_{x_1}(\mathbf{x}_{ij}^{[d]}),$$
$$A_{12ij} = \sum_{d=1}^{r} I_{x_1}(\mathbf{x}_{ij}^{[d]}) I_{x_2}(\mathbf{x}_{ij}^{[d]}),$$
$$A_{22ij} = \sum_{d=1}^{r} I_{x_2}(\mathbf{x}_{ij}^{[d]}) I_{x_2}(\mathbf{x}_{ij}^{[d]}),$$
$$B_{1ij} = \sum_{d=1}^{r} I_{x_1}(\mathbf{x}_{ij}^{[d]}) I_t(\mathbf{x}_{ij}^{[d]}),$$
$$B_{2ij} = \sum_{d=1}^{r} I_{x_1}(\mathbf{x}_{ij}^{[d]}) I_t(\mathbf{x}_{ij}^{[d]}).$$

Сгруппируем u_{1ij} и u_{2ij}:

$$\begin{cases} (A_{11ij} + \alpha)u_{1ij} + A_{12ij}u_{1ij} = \alpha(u_{1i+1j} + u_{1i-1j} + u_{1ij-1} + u_{1ij-1}) - B_{1ij} \\ A_{12ij}u_{2ij} + (A_{22ij} + \alpha)u_{2ij} = \alpha(u_{2i+1j} + u_{2i-1j} + u_{2ij-1} + u_{2ij-1}) - B_{2ij} \end{cases}.$$

Будем решать систему итерационным методом:

$$\begin{cases} (A_{11ij} + \alpha)u_{1ij}^{k} + A_{12ij}u_{1ij}^{k} = \alpha(u_{1i+1j}^{k-1} + u_{1i-1j}^{k-1} + u_{1ij-1}^{k-1} + u_{1ij-1}^{k-1}) - B_{1ij} \\ A_{12ij}u_{2ij}^{k} + (A_{22ij} + \alpha)u_{2ij}^{k} = \alpha(u_{2i+1j}^{k-1} + u_{2i-1j}^{k-1} + u_{2ij-1}^{k-1} + u_{2ij-1}^{k-1}) - B_{2ij} \end{cases}.$$

Пусть:

$$\widetilde{u}_1^{k-1} = u_{1i+1j}^{k-1} + u_{1i-1j}^{k-1} + u_{1ij-1}^{k-1} + u_{1ij-1}^{k-1}$$
$$\widetilde{u}_2^{k-1} = u_{2i+1j}^{k-1} + u_{2i-1j}^{k-1} + u_{2ij-1}^{k-1} + u_{2ij-1}^{k-1}$$

Тогда окончательное решение системы линейных уравнений запишется в виде:

$$\begin{cases} u_{1ij}^{k} = \widetilde{u}_1^{k-1} - (A_{11ij} + A_{22ij} + \alpha^2)^{-1}(A_{11ij}\widetilde{u}_1^{k-1} + A_{12ij}\widetilde{u}_2^{k-1} + B_{1ij}) \\ u_{2ij}^{k} = \widetilde{u}_2^{k-1} - (A_{11ij} + A_{22ij} + \alpha^2)^{-1}(A_{12ij}\widetilde{u}_1^{k-1} + A_{22ij}\widetilde{u}_2^{k-1} + B_{2ij}) \end{cases}.$$

При $k = 0$ u_{1ij} и u_{2ij} полагаются равным начальному приближению, рационально брать это приближение равным нулю или равным результату, полученным для предыдущего момента t.

Можно сделать следующее обобщение: считать, что r условий

$$I(\mathbf{x}^{[d]} + \mathbf{u}) - I(\mathbf{x}^{[d]}) \xrightarrow{\mathbf{u}} 0, \mathbf{x}^{[d]}, \mathbf{u} \in \Omega_{t_0}.$$

имеют разные веса $\omega^{[d]}$, тогда задача минимизации запишется следующим образом:

$$\int_{\Omega_{t_0}} \sum_{d=1}^{r} \omega^{[d]} \left(\left[I(\mathbf{x}^{[d]} + \mathbf{u}) - I(\mathbf{x}^{[d]}) \right]^2 \right) dx_1 dx_2 +$$
$$+\alpha \int_{\Omega_{t_0}} (\| \bigtriangledown u_1 \|^2 + \| \bigtriangledown u_2 \|^2) dx_1 dx_2 \xrightarrow{\mathbf{u}} 0.$$

При дискретизации изменится только способ расчета коэффициентов системы:

$$\begin{aligned}
A_{11ij} &= \sum_{d=1}^{r} \omega^{[d]} I_{x_1}(\mathbf{x}_{ij}^{[d]}) I_{x_1}(\mathbf{x}_{ij}^{[d]}), \\
A_{12ij} &= \sum_{d=1}^{r} \omega^{[d]} I_{x_1}(\mathbf{x}_{ij}^{[d]}) I_{x_2}(\mathbf{x}_{ij}^{[d]}), \\
A_{22ij} &= \sum_{d=1}^{r} \omega^{[d]} I_{x_2}(\mathbf{x}_{ij}^{[d]}) I_{x_2}(\mathbf{x}_{ij}^{[d]}), \\
B_{1ij} &= \sum_{d=1}^{r} \omega^{[d]} I_{x_1}(\mathbf{x}_{ij}^{[d]}) I_t(\mathbf{x}_{ij}^{[d]}), \\
B_{2ij} &= \sum_{d=1}^{r} \omega^{[d]} I_{x_1}(\mathbf{x}_{ij}^{[d]}) I_t(\mathbf{x}_{ij}^{[d]}).
\end{aligned}$$

Для определенности будем полагать, что

$$\sum_{d=1}^{r} \omega^{[d]} = 1$$

Эти коэффициенты можно выбирать различными способами. Самый простой $\omega^{[d]} = 1/r$. Если точки $\mathbf{x}^{[d]}$ лежат в окрестности точки $\mathbf{x}$, то более оправдано брать для точек, ближних к $\mathbf{x}$ большие веса, а для дальних меньшие.

На сетке, соответствующей пикселям изображения множество точек $\left\{ \mathbf{x}_{ij}^{[d]}, d = 1...r \right\}$ для точки $\mathbf{x}_{ij}$ можно выбрать следующим образом:

$$\{ \mathbf{x}_{nm}, n = -1...1, m = -1...1 \}$$

тогда формулы расчета коэффициентов системы линейных уравнений примут вид:

$$\begin{aligned}
A_{11ij} &= \sum_{n=-1,m=-1}^{1,1} I_{x_1}(\mathbf{x}_{i+n,j+m}) I_{x_1}(\mathbf{x}_{i+n,j+m}), \\
A_{12ij} &= \sum_{n=-1,m=-1}^{1,1} I_{x_1}(\mathbf{x}_{i+n,j+m}) I_{x_2}(\mathbf{x}_{i+n,j+m}), \\
A_{22ij} &= \sum_{n=-1,m=-1}^{1,1} I_{x_2}(\mathbf{x}_{i+n,j+m}) I_{x_2}(\mathbf{x}_{i+n,j+m}), \\
B_{1ij} &= \sum_{n=-1,m=-1}^{1,1} I_{x_1}(\mathbf{x}_{i+n,j+m}) I_t(\mathbf{x}_{i+n,j+m}), \\
B_{2ij} &= \sum_{n=-1,m=-1}^{1,1} I_{x_1}(\mathbf{x}_{i+n,j+m}) I_t(\mathbf{x}_{i+n,j+m}).
\end{aligned}$$

§2 Модификация для выделения границ

Приведем математическое описание предлагаемой модификации для выделения границ. Условие гладкости потока

$$\| \bigtriangledown u_1\|^2 + \| \bigtriangledown u_2\|^2 \stackrel{\mathbf{u}}{\to} 0, \mathbf{u} \in \Omega_{t_0}.$$

позволяет в последствии получить в итерационном подходе невырожденную систему линейных уравнений. Но для этого требуется гладкость потока, что приводит к сильному размытию границ движущихся объектов на изображении.

Рассмотрим задачу минимизации:

$$\int_{\Omega_{t_0}} ([I(\mathbf{x}+\mathbf{u}) - I(\mathbf{x})]^2) dx_1 dx_2 + \alpha \int_{\Omega_{t_0}} (\| \bigtriangledown u_1\|^2 + \| \bigtriangledown u_2\|^2) dx_1 dx_2 \stackrel{\mathbf{u}}{\to} 0.$$

Ей соответствует система дифференциальных уравнений

$$\begin{cases} I_{x_1}(\mathbf{x})[I_t(\mathbf{x}) + I_{x_1}(\mathbf{x})u_1 + I_{x_2}(\mathbf{x})u_2] - \alpha \bigtriangledown^2 u_1 = 0 \\ I_{x_2}(\mathbf{x})[I_t(\mathbf{x}) + I_{x_1}(\mathbf{x})u_1 + I_{x_2}(\mathbf{x})u_2] - \alpha \bigtriangledown^2 u_2 = 0 \end{cases}.$$

Для перехода к системе линейных уравнений в описанном ранее подходе использовались следующие формулы раскрытия лапласианов:

$$\bigtriangledown^2 u_{1ij} \approx u_{1i+1j} + u_{1i-1j} + u_{1ij-1} + u_{1ij-1} - 4u_{1ij}$$

$$\bigtriangledown^2 u_{2ij} \approx u_{2i+1j} + u_{2i-1j} + u_{2ij-1} + u_{2ij-1} - 4u_{2ij}$$

Модифицируем эти формулы. Будем считать, что для каждой точки известно являются ли соседние к ней точки граничные, т.е. известны отображения для определения наличия границы у конкретной точки:

$$Gu = \{\Omega_{t_0} \to 0, 1\} - \text{для границы сверху},$$

$$Gd = \{\Omega_{t_0} \to 0, 1\} - \text{для границы снизу},$$

$$Gl = \{\Omega_{t_0} \to 0, 1\} - \text{для границы слева},$$

$$Gr = \{\Omega_{t_0} \to 0, 1\} - \text{для границы справа.}$$

Значение 1 соответствует тому, что граница есть, 0 тому, что ее нет. Теперь можно модифицировать формулы раскрытия лапласианов следующим образом:

$$\bigtriangledown^2 u_{1ij} \approx Gd(\mathbf{x}_{ij})u_{1i+1j} + Gu(\mathbf{x}_{ij})u_{1i-1j} + Gl(\mathbf{x}_{ij})u_{1ij-1} + Gr(\mathbf{x}_{ij})u_{1ij-1} - \\ -(Gd(\mathbf{x}_{ij}) + Gu(\mathbf{x}_{ij}) + Gl(\mathbf{x}_{ij}) + Gr(\mathbf{x}_{ij}))u_{1ij}$$

$$\bigtriangledown^2 u_{2ij} \approx Gd(\mathbf{x}_{ij})u_{2i+1j} + Gu(\mathbf{x}_{ij})u_{2i-1j} + Gl(\mathbf{x}_{ij})u_{2ij-1} + Gr(\mathbf{x}_{ij})u_{2ij-1} - \\ -(Gd(\mathbf{x}_{ij}) + Gu(\mathbf{x}_{ij}) + Gl(\mathbf{x}_{ij}) + Gr(\mathbf{x}_{ij}))u_{2ij}$$

если для краткости обозначить $Gd(\mathbf{x}_{ij}) = Gd_{ij}$, $Gd(\mathbf{x}_{ij}) = Gu_{ij}$, $Gd(\mathbf{x}_{ij}) = Gl_{ij}$ и $Gr(\mathbf{x}_{ij}) = Gr_{ij}$, а их сумму

$$Gs_{ij} = Gd_{ij} + Gu_{ij} + Gl_{ij} + Gr_{ij}$$

можно нормировать $Gs_{ij} = 1$ то формулы раскрытия лапласианов примут вид

$$\bigtriangledown^2 u_{1ij} \approx Gd_{ij}u_{1i+1j} + Gu_{ij}u_{1i-1j} + Gl_{ij}u_{1ij-1} + Gr_{ij}u_{1ij-1} - u_{1ij}$$

$$\bigtriangledown^2 u_{2ij} \approx Gd_{ij}u_{2i+1j} + Gu_{ij}u_{2i-1j} + Gl_{ij}u_{2ij-1} + Gr_{ij}u_{2ij-1} - u_{2ij}$$

В результате значения $\bigtriangledown^2 u_{1ij}$ и $\bigtriangledown^2 u_{2ij}$ будут не равны нулю, если не равны нулю одновременно Gd_{ij}, Gu_{ij}, Gl_{ij}, Gr_{ij}. Эти значения можно вычислить для определенного момента времени t заранее и использовать в итерационном решении системы

$$\widetilde{u}^{k-1}_{1ij} = Gd_{ij}u^{k-1}_{1i+1j} + Gu_{ij}u^{k-1}_{1i-1j} + Gl_{ij}u^{k-1}_{1ij-1} + Gr_{ij}u^{k-1}_{1ij-1}$$

$$\widetilde{u}^{k-1}_{2ij} = Gd_{ij}u^{k-1}_{2i+1j} + Gu_{ij}u^{k-1}_{2i-1j} + Gl_{ij}u^{k-1}_{2ij-1} + Gr_{ij}u^{k-1}_{2ij-1}$$

$$\begin{cases} u^k_{1ij} = \widetilde{u}^{k-1}_1 - (A_{11ij} + A_{22ij} + \alpha^2)^{-1}(A_{11ij}\widetilde{u}^{k-1}_1 + A_{12ij}\widetilde{u}^{k-1}_2 + B_{1ij}) \\ u^k_{2ij} = \widetilde{u}^{k-1}_2 - (A_{11ij} + A_{22ij} + \alpha^2)^{-1}(A_{12ij}\widetilde{u}^{k-1}_1 + A_{22ij}\widetilde{u}^{k-1}_2 + B_{2ij}) \end{cases}.$$

§3 Система линейных уравнений

Выпишем систему уравнений для u

$$\begin{cases} B_{1ij} + A_{11ij}u_{1ij} + A_{12ij}u_{2ij} - \alpha \bigtriangledown^2 u_{1ij} = 0 \\ B_{2ij} + A_{12ij}u_{1ij} + A_{22ij}u_{2ij} - \alpha \bigtriangledown^2 u_{2ij} = 0 \end{cases}.$$

рассмотрим способ раскрытия лапласианов, введенный в предыдущем параграфе

$$\begin{cases} B_{1ij} + A_{11ij}u_{1ij} + A_{12ij}u_{2ij} - \\ -\alpha(Gd_{ij}u_{1i+1j} + Gu_{ij}u_{1i-1j} + Gl_{ij}u_{1ij-1} + Gr_{ij}u_{1ij-1} - Gs_{ij}u_{1ij}) = 0 \\ B_{2ij} + A_{12ij}u_{1ij} + A_{22ij}u_{2ij} - \\ -\alpha(Gd_{ij}u_{2i+1j} + Gu_{ij}u_{2i-1j} + Gl_{ij}u_{2ij-1} + Gr_{ij}u_{2ij-1} - Gs_{ij}u_{2ij}) = 0 \end{cases}.$$

Таким образом мы пришли системе линейных уравнений порядка $2 * w * h$. Обозначим эту систему уравнений как $H_0 z_0 = q_0$. В случае $w = h = 4$ схема матрицы системы

$$H_0 = \begin{pmatrix} A & \vdots & B \\ \dots & & \dots \\ B & \vdots & C \end{pmatrix}, A = A_{11ij}, B = A_{12ij}, C = A_{22ij}, i, j = \overline{1, wh}$$

при первом способе раскрытия Лапласиана имеет вид, показанный на рисунке 4.

Первые $w * h$ уравнений отвечают u_{1ij}, вторые $w * h$ уравнений отвечают u_{2ij}. На схеме нулям соответствуют незакрашенные цвета, сплошной штриховке отвечают $a_{i,i} = A_{11ij} + Gs_{ij}\alpha$ и $c_{i,i} = A_{22ij} + Gs_{ij}\alpha$ в верхней левой и правой нижней четверти соответственно. Диагональной штриховке отвечает элемент $-G_{ij}\alpha$, G_{ij} - один из Gd_{ij},Gu_{ij},Gl_{ij},Gr_{ij}, а кругам элемент $b_{i,i} = A_{12ij}$ Вектор свободных членов будет иметь вид

$$q_0 = (-B_{111}, \dots, -B_{1wh}, -B_{211}, \dots, -B_{2wh})^T.$$

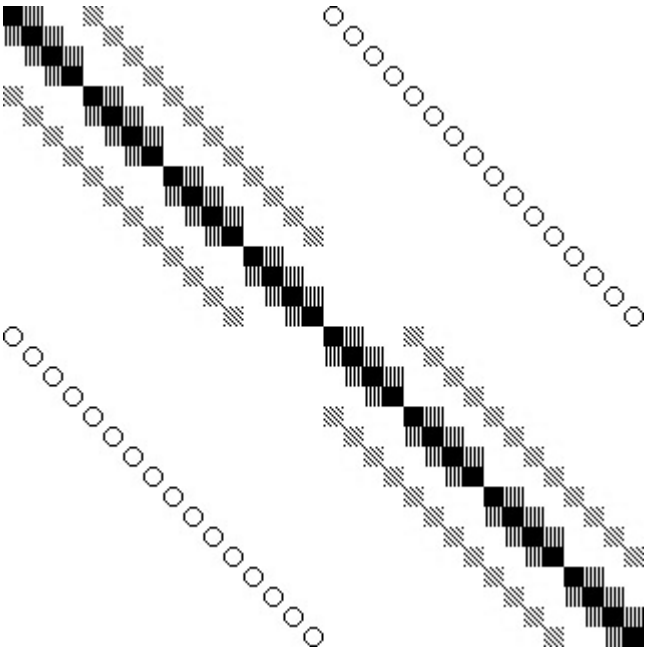

Рис. 4: Схема матрицы.

Искомый вектор:

$$z_0 = (u_{111}, \ldots, u_{1wh}, u_{211}, \ldots, u_{2wh})^T.$$

Уравнения и переменные можно перегруппировать следующим образом $Hz = q$:

$$H = \{H_{i,j}\}, i \in \overline{1, wh}, j \in \overline{1, wh}$$

$$H_{i,j} = \begin{pmatrix} a_{ij} & b_{ij} \\ b_{ij} & c_{ij} \end{pmatrix},$$

положим $n = wh$, тогда

$$z = \left((u_{111}, u_{211})^T, \ldots, (u_{1wh}, u_{2wh})^T\right)^T = (z_1, \ldots, z_n)^T,$$

$$q = \left((-B_{111}, -B_{211})^T, \ldots, (-B_{1wh}, -B_{2wh})^T\right)^T = (q_1, \ldots, q_n)^T.$$

§4 Численные методы решения систем линейных уравнений

Для решения системы $Hz = q$ можно использовать блочные численные методы [11-14], построенные на основе известных численных методов Якоби, Гаусса-Зейделя и метода последовательной верхней релаксации.

Метод Якоби для данной задачи примет вид:

$$H_{ii}z_i^{k+1} = -\sum_{j\neq i, j=1}^{n} H_{ij}z_j^k + q_i, i = \overline{1,n}, k = \overline{0,\infty}, z^0 = q,$$

погрешность z^k относительно точного решения z^* определяется формулой

$$||z^k - z^*|| < Q^k||z^0 - z^8||, \text{ где } Q = ||H||$$

или с помощью нормы невязки

$$||r^k|| = ||q - Hz^k|| \leq \epsilon.$$

Метод Гаусса-Зейделя определяется формулой

$$H_{ii}z_i^{k+1} = -\sum_{j<i, j=1}^{n} H_{ij}z_j^{k+1} - \sum_{j>i, j=1}^{n} H_{ij}z_j^k + q_i, i = \overline{1,n}, k = \overline{0,\infty}, z^0 = q,$$

погрешность:

$$||z^k - z^*|| \leq \frac{Q_r}{1-Q}||z^k - z^{k-1}||, \text{ где } Q = ||H||, Q_r = ||H_*||, H_* = H^{-1}H_r,$$

при разложении матрицы на диагональную, верхнюю и нижнюю треугольную $H = H_L + H_D + H_r$, или с помощью нормы невязки

$$||r^k|| = ||q - Hz^k|| \leq \epsilon.$$

Метод последовательной верхней релаксации:

$$H_{ii}z_i^{k+1} = \omega\left(-\sum_{j<i, j=1}^{n} H_{ij}z_j^{k+1} - \sum_{j>i, j=1}^{n} H_{ij}z_j^k + q_i\right) + (1-\omega)H_{ii}z_i^k, i = \overline{1,n}, k = \overline{0,\infty},$$

Погрешность

$$||r^k|| = ||q - Hz^k|| \leq \epsilon.$$

§5 Сходимость блочных методов

Будем рассматривать блочную систему, введенную в третьем параграфе этой главы $Hz = q$.

Для доказательства сходимости блочных итерационных методов Якоби, Гаусса-Зейделя и метода последовательной верхней релаксации применительно к этой системе будем пользоваться следующими условиями сходимости [15]:

1. $a_{ii}c_{ii} - b_{ii}^2 > 0, a_{ii} > 0, c_{ii} > 0, i = \overline{1,n}$

2. $\frac{a_{ii}+c_{ii}}{2} \geq \sum_{r=1,r\neq i}^{n} ||a_{ir}|| + \sqrt{\left(\frac{a_{ii}-c_{ii}}{2}\right)^2 + b_{ii}^2}$ и для некоторого i это неравенство выполняется в строгом виде

3. матрица системы является блочно неприводимой

В [15] доказывается справедливость условий 1. и 2. для стандартной системы, которая используется в стандартном случае алгоритма Horn-Schunck. В обоих случаях матрицы системы линейных уравнений будут блочно неприводимыми.

Приведем доказательство справедливости условий 1. и 2. для нашего смешенного алгоритма. Для упрощения выкладок положим

$$I_x(\mathbf{x}_{ij}) = U_{ij}, I_y(\mathbf{x}_{ij}) = V_{ij}, i, j \in 1, 2.$$

Условие 1.

$$a_{ii}c_{ii} - b_{ii}^2 = \left(\sum_{i,j=1}^{2} U_{ij}^2 + Gs_{ij}\alpha\right)\left(\sum_{i,j=1}^{2} V_{ij}^2 + Gs_{ij}\alpha\right) - \left(\sum_{i,j=1}^{2} U_{ij}V_{ij}\right)^2 =$$

$$= \sum_{i,j=1}^{2} U_{ij}^2 Gs_{ij}\alpha + \sum_{i,j=1}^{2} V_{ij}^2 Gs_{ij}\alpha + Gs_{ij}^2\alpha^2 + \sum_{i,j=1}^{2} U_{ij}^2 \sum_{i,j=1}^{2} V_{ij}^2 -- \left(\sum_{i,j=1}^{2} U_{ij}V_{ij}\right)^2 >$$

$$> \sum_{i,j=1}^{2} U_{ij}^2 \sum_{i,j=1}^{2} V_{ij}^2 - \left(\sum_{i,j=1}^{2} U_{ij}V_{ij}\right)^2 =$$

$$= (U_{11}^2+U_{21}^2+U_{12}^2+U_{22}^2)(V_{11}^2+V_{21}^2+V_{12}^2+V_{22}^2) - (U_{11}V_{11}+U_{21}V_{21}+U_{12}V_{12}+U_{22}V_{22}) =$$

$$= (U_{11}V_{12} - U_{12}V_{11})^2 + (U_{11}V_{21} - U_{21}V_{11})^2 + (U_{11}V_{22} - U_{22}V_{11})^2 +$$

$$+(U_{12}V_{21} - U_{21}V_{12})^2 + (U_{12}V_{22} - U_{22}V_{12})^2 + (U_{21}V_{22} - U_{22}V_{21})^2 \geq 0,$$

$$a_{ii} = \sum_{i,j=1}^{2} U_{ij}^2 + Gs_{ij}\alpha > 0, c_{ii} = \sum_{i,j=1}^{2} V_{ij}^2 + Gs_{ij}\alpha > 0.$$

Условие 2.

$$\frac{a_{ii} + c_{ii}}{2} \geq \sum_{r=1,r\neq i}^{n} ||a_{ir}|| + \sqrt{\left(\frac{a_{ii} - c_{ii}}{2}\right)^2 + b_{ii}^2}$$

Для точек, расположенных на границе изображения $\sum_{r=1,r\neq i}^{n} ||a_{ir}|| < Gs_{ij}\alpha$, что будет обеспечивать случай строгого выполнения неравенства в условии, и $\sum_{r=1,r\neq i}^{n} ||a_{ir}|| < Gs_{ij}\alpha$ для точек, расположенных внутри изображения.

$$Gs_{ij}\alpha + \frac{\sum_{i,j=1}^{2} U_{ij}^2 + \sum_{i,j=1}^{2} V_{ij}^2}{2} \geq$$

$$\geq Gs_{ij}\alpha + \sqrt{\left(\frac{\sum_{i,j=1}^{2} U_{ij}^2 - \sum_{i,j=1}^{2} V_{ij}^2}{2}\right)^2 + \left(\sum_{i,j=1}^{2} U_{ij}V_{ij}\right)^2}$$

$$\left(\frac{\sum_{i,j=1}^{2} U_{ij}^2 + \sum_{i,j=1}^{2} V_{ij}^2}{2}\right)^2 \geq \left(\frac{\sum_{i,j=1}^{2} U_{ij}^2 - \sum_{i,j=1}^{2} V_{ij}^2}{2}\right)^2 + \left(\sum_{i,j=1}^{2} U_{ij}V_{ij}\right)^2$$

$$\left(\frac{\sum_{i,j=1}^{2} U_{ij}^2 + \sum_{i,j=1}^{2} V_{ij}^2}{2}\right)^2 - \left(\frac{\sum_{i,j=1}^{2} U_{ij}^2 - \sum_{i,j=1}^{2} V_{ij}^2}{2}\right)^2 \geq \left(\sum_{i,j=1}^{2} U_{ij}V_{ij}\right)^2$$

$\sum_{i,j=1}^{2} U_{ij}^2 \sum_{i,j=1}^{2} V_{ij}^2 \geq \left(\sum_{i,j=1}^{2} U_{ij}V_{ij}\right)^2$, что уже рассматривалось в условии 1.

Конец доказательства.

Глава 3 Программная реализация вычисления оптического потока

§1 Общая схема реализации вычислительной системы

В данной главе приводится описание построенного прототипа распределенной системы для распознавания движения на видео или последовательности изображений [16].

Прототип системы реализован в виде веб-сервиса. Общая схема вычислительного процесса представлена на рис. 6. Реализованный сервис предоставляет возможность вычисления поля скоростей для набора изображений или кадров видиофайла.

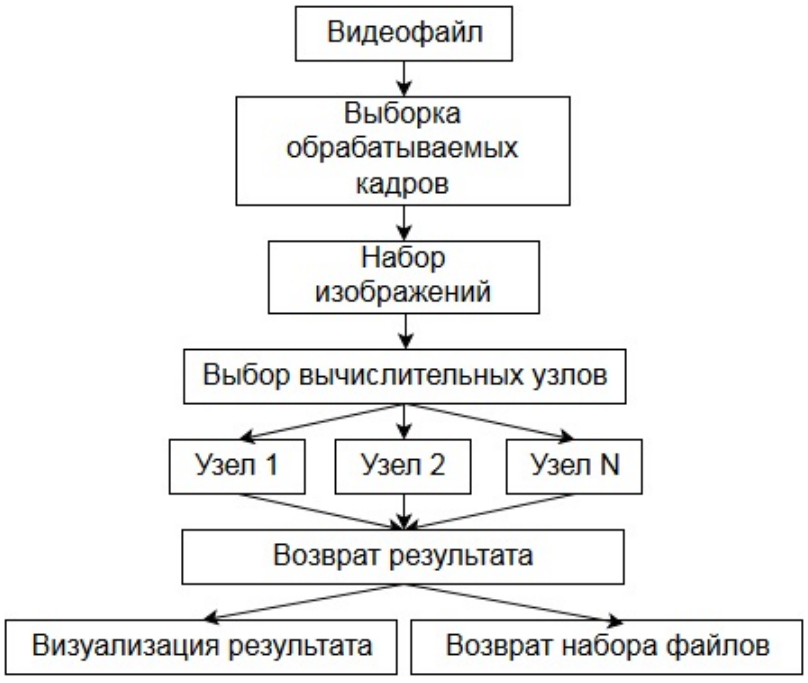

Рис. 5: Общая схема вычислительного процесса.

Пользователь на вход подает несколько изображений или видеофайл, во втором случае по указаным моментам времени из видеофайла извлекаются изображения. Для каждой пары из набора изображений необходимо

вычислить поле скоростей, матрицу проекций скоростей по осям x и y. Запрос пользователя представляет собой отдельную задачу. Процесс вычисления состоит из следующих этапов:

1. В случае обработки видеофайла, извлечение по кадров из видео по указанным пользователем временным меткам.

2. Выбор системой балансировки вычислительных узлов для решения задачи.

3. Загрузка данных на вычислительные узелы.

4. Вычисление поля скоростей для задачи.

5. Предоставление результата пользователю в виде набора файлов. Для каждой пары изображений предоставляется отдельный файл.

6. Пользователь может визуализировать полученный результат средствами, предоставляемыми на сайте.

§2 Средства реализации

Для реализации веб-сервиса были выбраны следующие средства.

Heroku

Облачный сервис Heroku [17] предоставляет вычислительные ресурсы для размещения проектов, созданных на базе различных языков программирования: PHP, Java, Ruby, Python и некоторые другие. Каждый проект запускается в отдельном контейнере, что обеспечивает его защищенность и легкую масштабируемость.

В описываемой реализаци используется вариант однопоточных проектов для организации вычислительных сервисов, каждый из которых может вычислять поле скоростей для набора изображений, и сервиса-контроллера, который будет предоставлять информацию о состоянии вычислительных сервисов. Отдельный сервис служит для выделения кадров из видеофайла. Все они реализованы с использованием языка программирования JAVA.

Доступ к вычислительным сервисам (вычислительные узлы) организуется через POST запрос.

Heroku DB

Кроме вычислительных ресурсов облачный сервис Heroku предоставляет возможность размещения базы данных под СУБД PostgreSQL, эта возможность используется для хранения состояний вычислительных узлов. Каждый, из них, в начале вычисления задачи отмечет в базе то, что он занят. При завершении вычисления задачи он отмечает то, что он освободился. Схема базы данных состояний вычислительных узлов представлена на рис. 6.

Столбцы таблицы nodes имеют следующее назначение:

id - номер вычислительного узла 1 - 8

isused - занятость вычислительного узла true/false

instack - дополнительное поле, не используется

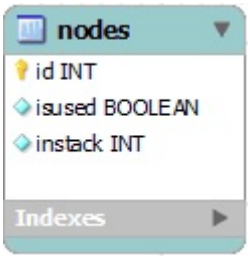

Рис. 6: Схема базы данных состояний вычислительных узлов.

Файловый сервер и веб-сервер

Основной сайт, через который предоставляется доступ к вычислительной системе представляет собой отдельный сервис, реализованный на PHP, также развернутый на Heroku. Временное хранилище изображений, полученных от пользователей, и результаты вычислений размещаются на ftp сервисе byethost.

Визуализация на стороне клиента

Пользователь системы может визуализировать результат с помощью скрипта javascript в браузере.

§3 Описание жизненного цикла задачи

Схема процесса обработки отдельной задачи представлена на рис. 7. Пользователь выбирает на основном сайте набор изображений и запускает процесс обработки задачи.

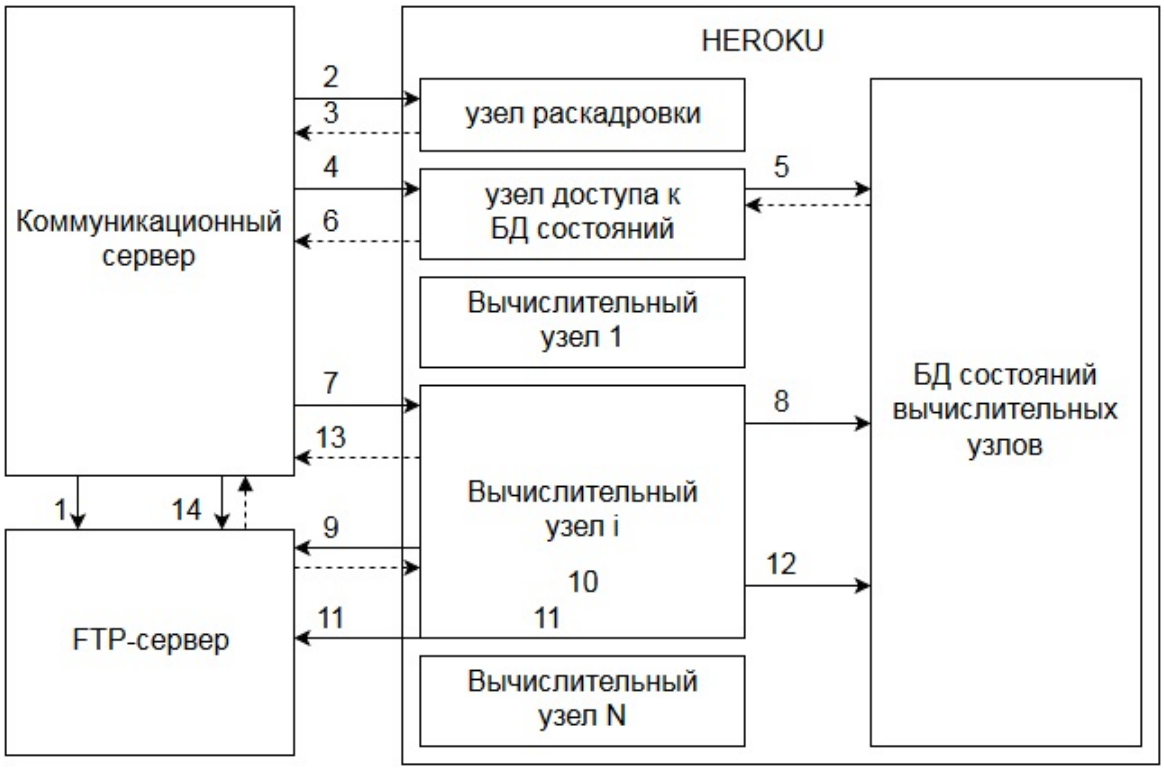

Рис. 7: Схема цикла обработки последовательности изображений.

В случае обработки видео предварительно происходит извлечение необходимых кадров, которые представляют собой последовательность изображений.

Далее приведено описание шагов обработки задачи:

1. Коммуникационный сервер загружает видео или изображения на ftp сервер.

2. В случае обработки видео, оно загружается на узел для раскадровки, дополнительно

3. Результат раскадровки сохраняется на ftp-сервер, коммуникацион-

ному серверу возвращаются ссылки на имена файлов изображений подлежащих обработке.

4. Коммуникационный сервер обращается к узлу-контроллеру (heroku) для получения информации о состоянии вычислительных узлов.

5. Узел-контроллер соединяется с базой данных состояний вычислительных узлов (Heroku postgresql) и загружает информацию о состоянии системы.

6. Узел-контроллер возвращает коммуникационному серверу информацию о состоянии вычислительных узлов в виде строки

```
1,false;2,true;3,false;4,false;
5,true;6,false;7,false;8,false;
```

7. Коммуникационный сервер выбирает узелs, на которыt подается запрос на обработку с параметрами с параметрами count - количество изображений в последовательности, array - массив имен файлов.

8. Вычислительный узел указывает в базе данных состояний узлов Heroku postgresql информацию о том, что он занят.

9. По этим ссылкам вычислительный узел загружает изображения из ftp-сервера.

10. Вычислительный узел производит решение задачи: для каждой пары изображений вычисляет поле скоростей в виде пары матриц в одном файле данных.

11. Вычислительный узел сохраняет файлы результата на ftp-сервер.

12. Вычислительный узел указывает в базе данных состояний узлов Heroku postgresql информацию о том, что он свободен.

13. Коммуникационный сервер получает ответ на POST запрос от вычислительного узла, содержащий информацию о завершении обработки задачи. Пользователю предоставляется результат в виде набора ссылок на файлы результата, хранящиеся на ftp сервере.

14. Пользователь может скачать файлы результата с ftp-сервера или просмотреть визуализацию потока на сайте.

Протокол HTTP используется при взаимодействии коммуникационного сервера с узлами Heroku и для общения с пользователем. Протокол FTP используется при загрузке и сохранении выйлов на ftp-сервер вычислительными узлами и узлом раскадровки.

Глава 4 Примеры и анализ работы алгоритмов

§1 Пример: движение прямоугольника

Продемонстрируем работу алгоритма на нескольких примерах.

Рассмотрим движение черного прямоугольника на белом фоне рис. 8. Движение происходит на единичный вектор вниз.

Рис. 8: Прямоугольник.

Результат работы стандартного алгоритма Horn Schunck показан на рисунке 9.

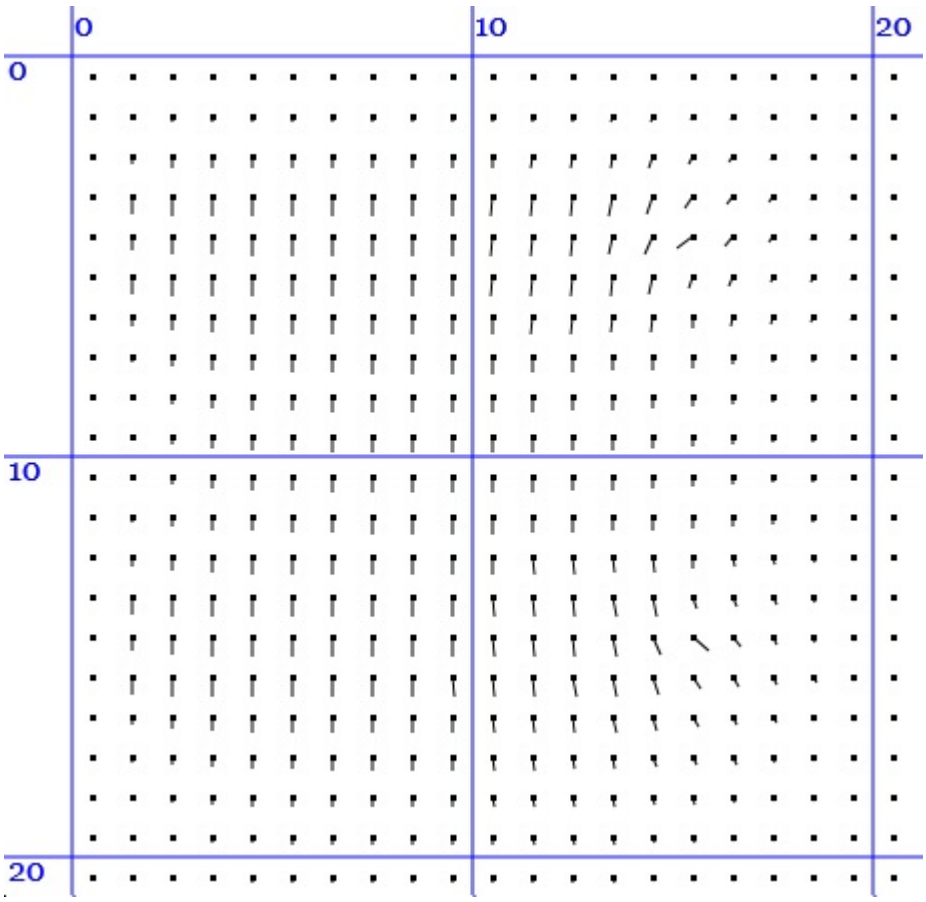

Рис. 9: Поток для прямоугольника расчитаный стандартным алгоритмом Horn Schunck. Цифрами отмечена координатная сетка пикселей на изображении.

Результат работы стандартного алгоритма Лукаса-Канаде показан на рисунке 10.

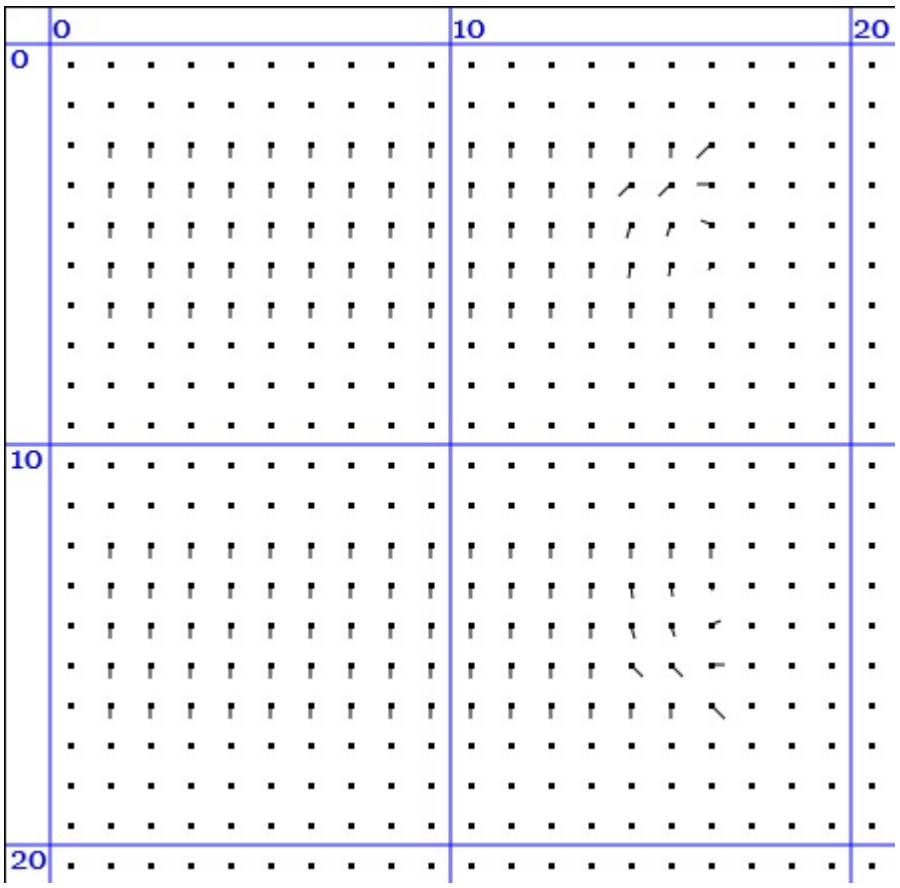

Рис. 10: Поток для прямоугольника расчитаный стандартным алгоритмом Лукаса-Канаде. Цифрами отмечена координатная сетка пикселей на изображении.

Результат работы модифицированного смешанного алгоритма Horn Schunck Лукаса-Канаде показан на рисунке 11.

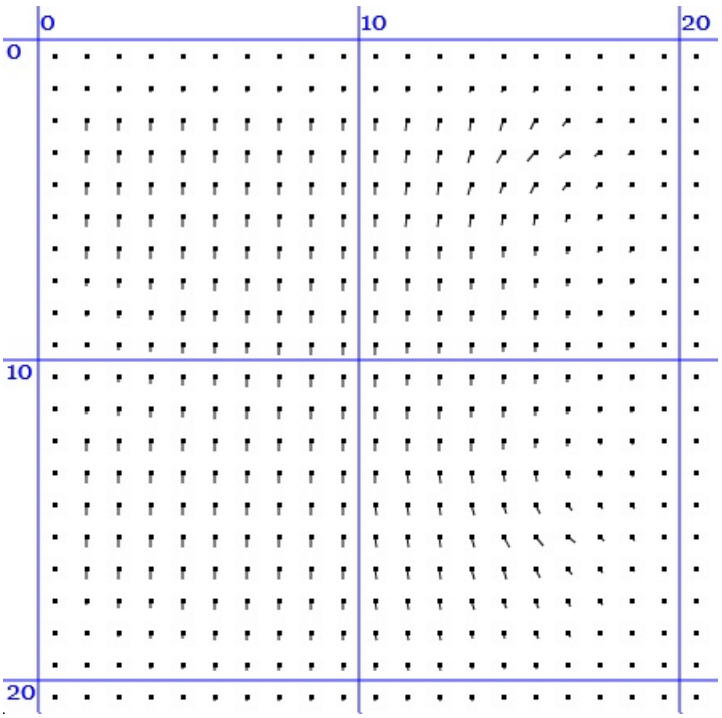

Рис. 11: Поток для прямоугольника расчитаный модифицированным смешанным алгоритмом Horn Schunck Лукаса-Канаде. Цифрами отмечена координатная сетка пикселей на изображении.

Результат работы модифицированного смешанного алгоритма Horn Schunck Лукаса-Канаде с модификацией выделения границ показан на рисунке 12.

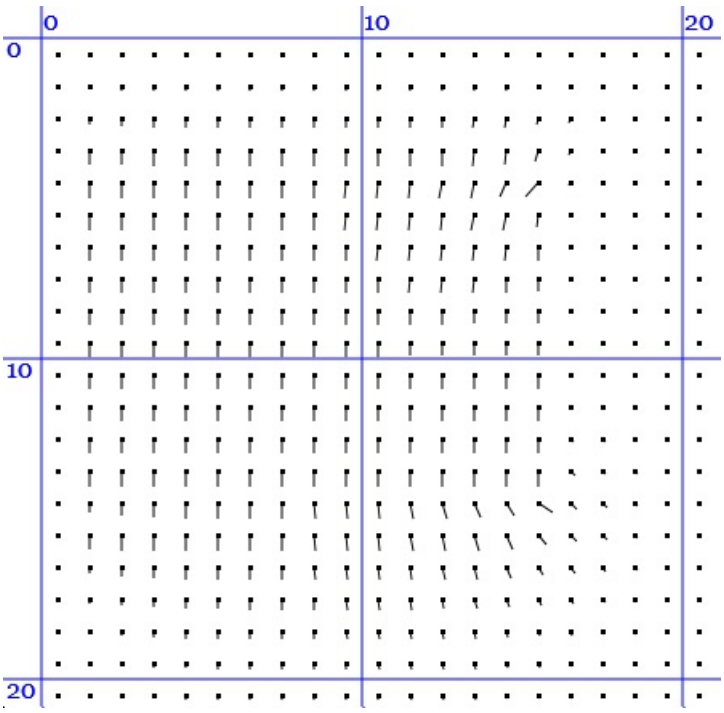

Рис. 12: Поток для прямоугольника расчитаный модифицированным смешанным алгоритмом Horn Schunck Лукаса-Канаде с модификацией выделения границ. Цифрами отмечена координатная сетка пикселей на изображении.

.

§2 Пример: сдвиг изображения

Рассмотрим сдвиг изображения на вектор $1, 1$ вниз-влево.

Рис. 13: Тестовое изображение. Красной рамкой выделена область, для которой будет показано поле скоростей.

Результат работы стандартного алгоритма Horn Schunck показан на рисунке 14.

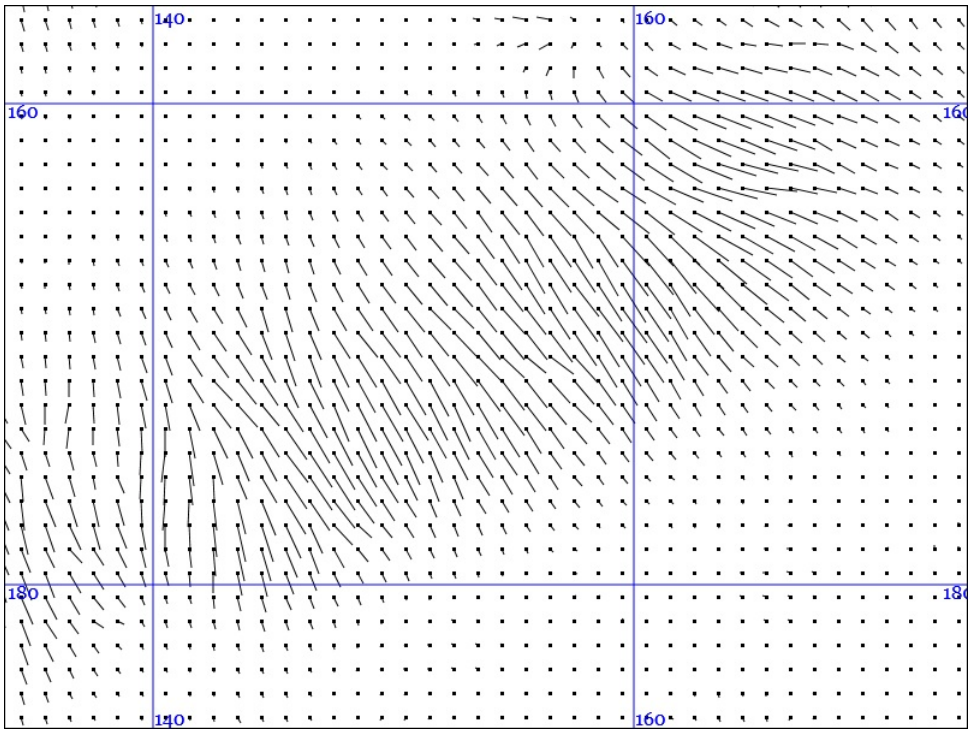

Рис. 14: Поток для прямоугольника расчитаный стандартным алгоритмом Horn Schunck. Цифрами отмечена координатная сетка пикселей на изображении.

Результат работы модифицированного смешанного алгоритма Horn Schunck Лукаса-Канаде показан на рисунке 15.

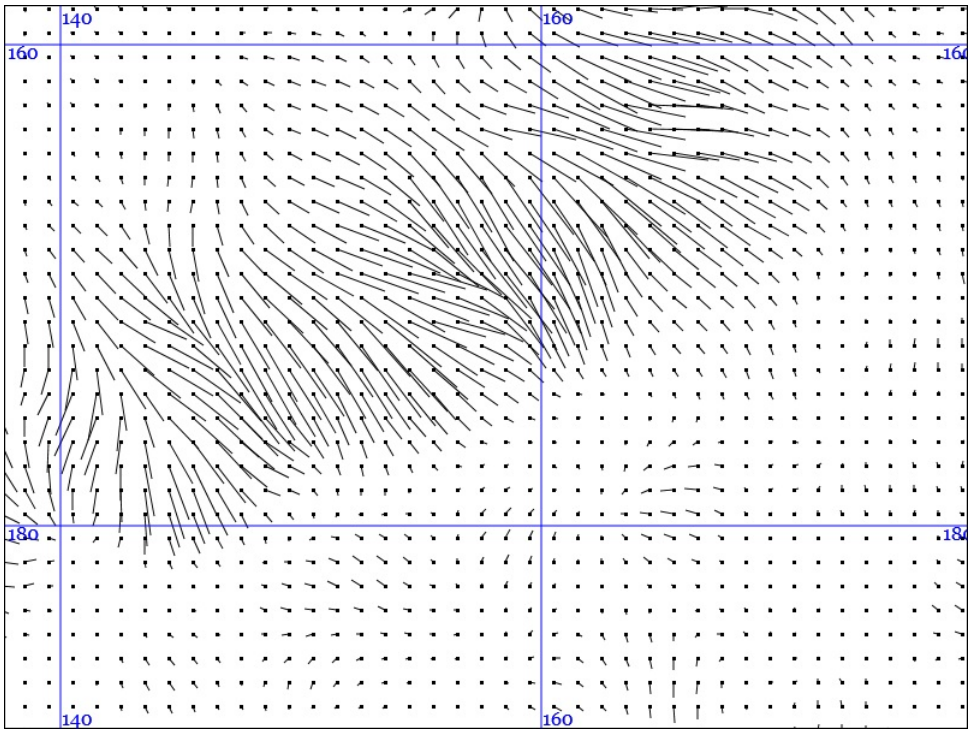

Рис. 15: Поток для прямоугольника расчитаный стандартным алгоритмом Horn Schunck. Цифрами отмечена координатная сетка пикселей на изображении.

§3 Пример: сдвиг карты

Метод оптического потока также можно применять для анализа карт местности, получаемых в результате аэрофотосьемки.

Рассмотрим сдвиг карты на вектор $1, 1$ вниз-влево.

Рис. 16: Тестовое изображение. Красной рамкой выделена область, для которой будет показано поле скоростей.

Результат работы стандартного алгоритма Horn Schunck показан на рисунке 17.

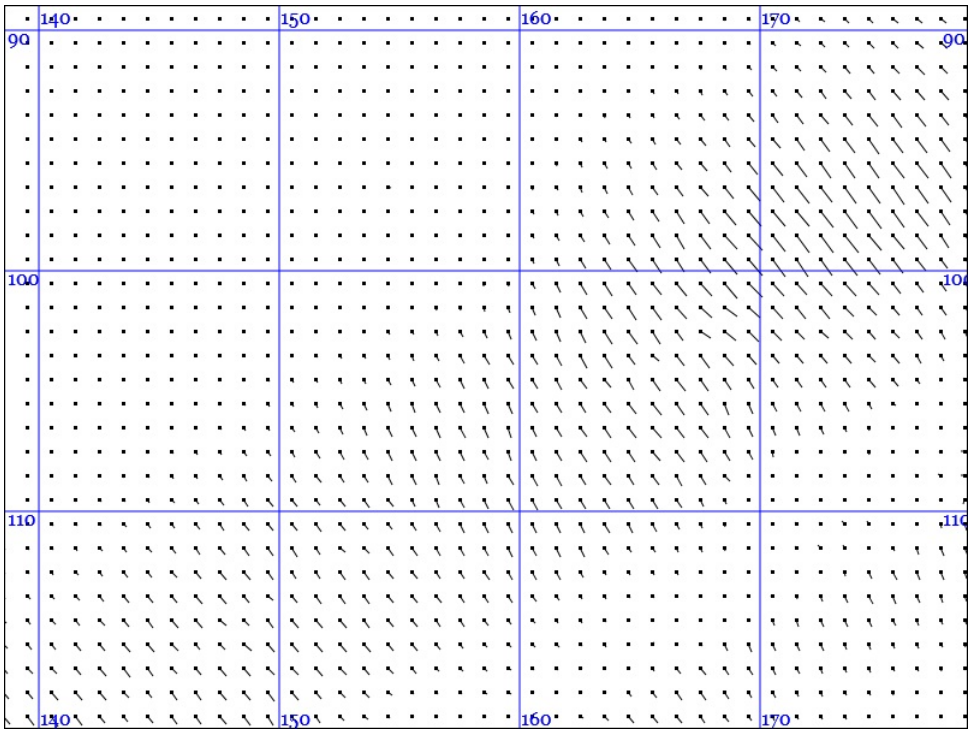

Рис. 17: Поток для прямоугольника расчитаный стандартным алгоритмом Horn Schunck. Цифрами отмечена координатная сетка пикселей на изображении.

Результат работы модифицированного смешанного алгоритма Horn Schunck Лукаса-Канаде показан на рисунке 18.

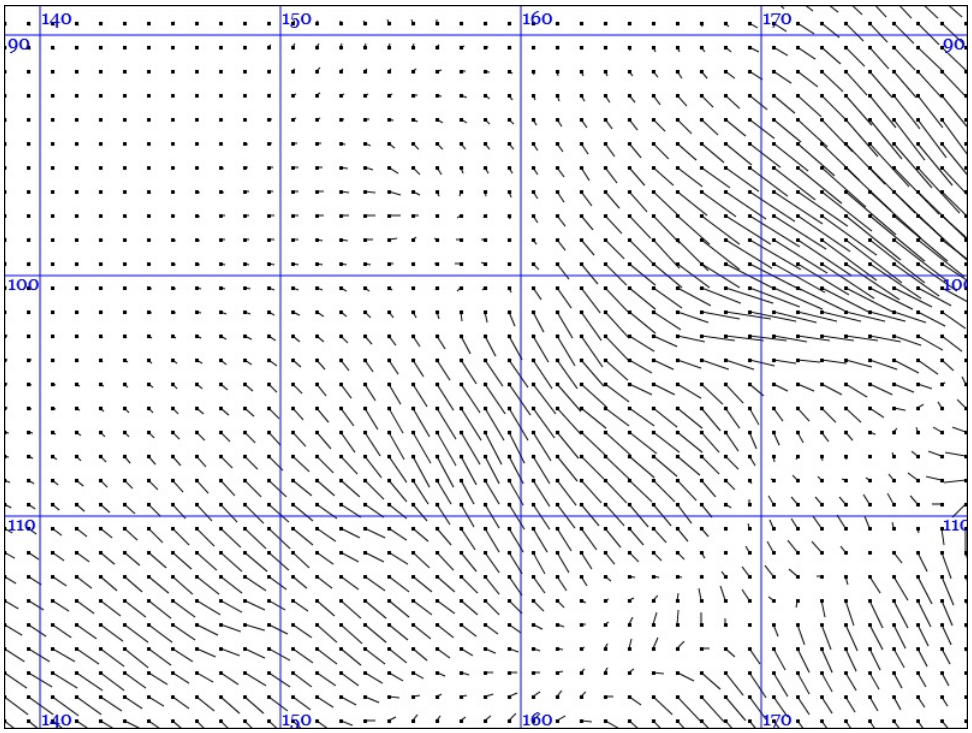

Рис. 18: Поток для прямоугольника расчитаный стандартным алгоритмом Horn Schunck. Цифрами отмечена координатная сетка пикселей на изображении.

§4 Анализ работы алгоритмов

Для проведения анализа работы алгоритмов необходимо выбрать величины, по которым производить сравнение. Ранее были описаны три примера, для каждого из них мы знаем заданные изначально векторы сдвига и определенные алгоритмами их оценки. В качестве таких величин выберем следующие величины, которые будут отражать средние отклонения вычисленных величин от реальных

$$E_\phi = \frac{1}{wh} \sum_{i,j=i_0,j_0}^{i_w,j_h} (\phi_{ij} - \phi_0)$$

$$E_r = \frac{1}{wh} \sum_{i,j=i_0,j_0}^{i_w,j_h} (u_{xij} - u_x)^2 + (u_{yij} - u_y)^2$$

здесь область $i = i_0...i_w, j = j_0...j_h$ фрагмент изображения размером $wh, 0, u_x, u_y$ - соответственно фазовый угол, и компоненты известного вектора сдвига изображения. $\phi_{ij}, u_{xij}, u_{yij}$ - соответственно фазовый угол, и компоненты вектора, вычисленного для точки i, j.

Далее для каждого примера приведем сравнение результатов работы следующих алгоритмов:

1. Орригинального алгоритма Horn Schunck,

2. Модифицированного смешанного алгоритма Horn Schunck и ЛукасаКанаде, где точке с координатами i_0, j_0 соответствует множество точек

$$\{(i_0 + i, j_0 + j) | i \in \{-1, 0, 1\}, j \in \{-1, 0, 1\}\}$$

3. Модифицированного смешанного алгоритма Horn Schunck и Лукаса-Канаде с подчеркиванием границ,

4. Орригинальный алгоритм Horn Schunck с подчеркиванием границ.

Результаты сравнения методов для примера сдвиг прямоугольника вниз представлены на таблице 1, для прмера сдвиг изображения вниз-вправо на таблице 2 и сдвига карты на таблице 3.

E_ϕ - средняя ошибка фазового угла вектора поля скоростей				
число итераций	5	20	40	60
оригинальный Horn-Shunck	1.568	1.548	1.545	1.544
смешанный Horn-Shunck и Лукаса-Канаде	1.562	1.540	1.539	1.537
смешанный Horn-Shunck и Лукаса-Канаде с подчеркиванием границ	1.565	1.550	1.547	1.543
оригинальный Horn-Shunck с подчеркиванием границ	1.525	1.468	1.461	1.458
E_r - средняя ошибка длины вектора поля скоростей				
число итераций	5	20	40	60
оригинальный Horn-Shunck	0.614	0.560	0.477	0.434
смешанный Horn-Shunck и Лукаса-Канаде	0.474	0.419	0.355	0.328
смешанный Horn-Shunck и Лукаса-Канаде с подчеркиванием границ	0.469	0.415	0.356	0.335
оригинальный Horn-Shunck с подчеркиванием границ	0.610	0.539	0.462	0.417

Таблица 1: Сравнение методов для сдвига прямоугольника

E_ϕ - средняя ошибка фазового угла вектора поля скоростей				
число итераций	5	20	40	60
оригинальный Horn-Shunck	0.875	0.595	0.577	0.566
смешанный Horn-Shunck и Лукаса-Канаде	0.658	0.607	0.568	0.538
смешанный Horn-Shunck и Лукаса-Канаде с подчеркиванием границ	0.658	0.607	0.568	0.538
оригинальный Horn-Shunck с подчеркиванием границ	0.875	0.566	0.595	0.577
E_r - средняя ошибка длины вектора поля скоростей				
число итераций	5	20	40	60
оригинальный Horn-Shunck	2.728	2.641	2.500	2.376
смешанный Horn-Shunck и Лукаса-Канаде	2.457	1.865	1.515	1.394
смешанный Horn-Shunck и Лукаса-Канаде с подчеркиванием границ	2.457	1.865	1.515	1.394
оригинальный Horn-Shunck с подчеркиванием границ	2.728	2.376	2.641	2.500

Таблица 2: Сравнение методов для сдвига изображения

E_ϕ - средняя ошибка фазового угла вектора поля скоростей				
число итераций	5	20	40	60
оригинальный Horn-Shunck	0.516	0.295	0.236	0.207
смешанный Horn-Shunck и Лукаса-Канаде	0.619	0.627	0.668	0.739
смешанный Horn-Shunck и Лукаса-Канаде с подчеркиванием границ	0.619	0.627	0.668	0.739
оригинальный Horn-Shunck с подчеркиванием границ	0.516	0.295	0.236	0.207
E_r - средняя ошибка длины вектора поля скоростей				
число итераций	5	20	40	60
оригинальный Horn-Shunck	2.637	2.498	2.347	2.246
смешанный Horn-Shunck и Лукаса-Канаде	2.268	1.877	1.795	1.578
смешанный Horn-Shunck и Лукаса-Канаде с подчеркиванием границ	2.268	1.877	1.795	1.578
оригинальный Horn-Shunck с подчеркиванием границ	2.637	2.498	2.347	2.246

Таблица 3: Сравнение методов для сдвига катры

Анализируя результаты можно сказать что смешанный алгоритм дает более точные оцеки поля скоростей, а подчеркивание границ эффективно не всегда.

Заключение

В работе было сделано следующее:

1. Был построен прототип системы распределенного вычисления поля скоростей по видео и последовательности изображений.

2. Было предложено две модификации метода вычисления поля скоростей Horn-Schunck: смешанный алгоритм Horn-Schunck и Лукаса-Канаде и модификация подчеркивания границ.

3. Для смешанного алгоритма Horn-Schunck и Лукаса-Канаде была показана сходимость итерационных методов Якоби, Гаусса-Зейделя и метода последовательной верхней релаксации.

4. На примерах было продемонстрировано, что смешанный алгоритм дает более эффективный результат, чем стандартный метода Horn-Schunck.

Таким образом, цель работы, заключавшаяся в построении системы распределенного вычисления поля скоростей, достигнута.

Построенный прототип системы можно улучшить несколькими способами: ускорить вычислительный процесс, добавив распределение одной задачи на несколько узлов, усовершенствовать применяемый алгоритм, усовершенствовать отказоустойчивость системы при выходе из строя вычислительных узлов.

Список литературы

[1] Гонсалес Дж.Ту.Р. принципы распознавания образов. М.: Мир, 1978, 414 с.

[2] Uijlings J. R. R. Selective search for object recognition// International Journal of Computer Vision. 2013. No. 104. P. 154–171.

[3] Bruce D. Lucas, and Takeo Kanade. An iterative image registration technique with an application to stereo vision// IJCAI, 1981, P. 674–679.

[4] Horn B. K. P., Schunck B. G. Determining Optical Flow // Artificial Intelligence. 1981. No 17. P. 185–203.

[5] Papenberg N. Highly Accurate Optic Flow Computation with Theoretically JustifiedWarping // International Journal of Computer Vision. 2006. Vol. 2, No 67. P. 141–158.

[6] Barron J.L., Fleet D.J. Beauchemin S.S. Performance of Optical Flow Techniques // International Journal of Computer Vision. 1994. No 12. P. 43–77.

[7] Fleet D.J., Weiss Y. Optical Flow Estimation // Mathematical models for Computer Vision. 2005. P. 25.

[8] Black M.J. Robust Incremential Optical Flow // Yale University. 1992. P.280.

[9] Anandan P. A computational framework and an algorithm for the measurement of visual motion // International Journal of Computer Vision. 1989. P. 283—310.

[10] Котина Е. Д., Пасечная Г. А., Определение поля скоростей в задачах

обработки изображений, Изв. Иркутского гос. ун-та. Сер. Математика, 2013, 48–59 с.

[11] Фаддеев Д. К., Фаддеева В. Н. Вычислительные методы линейной алгебры, Численные методы линейной алгебры. Параллельные вычисления, Л.: Наука 1975, 228 с.

[12] Форсайт Дж. Молер К. Численное решение систем линейных алгебраических уравнений М.: Мир, 1969, 167 с.

[13] Воеводин В. В. Параллельные вычисления, СПб: БХВ, 2002, 602 с.

[14] Вержбицкий В. М. Основы численных методов: Учебник для вузов/ В. М. Вержбицкий. — М.: Высш. шк., 2002. — 840 с.

[15] Котина Е. Д. О сходимости блочных итерационных методов / Е. Д. Котина// Изв. Иркут. гос. ун-та. – 2012. – Т. 5, вып. 3. – С. 41–55

[16] Электронный ресурс: http://velocityflow.herokuapp.com/ [дата обращения: 10.05.16]

[17] Электронный ресурс: https://www.heroku.com/ [дата обращения: 10.05.16]